UNIVERSIDAD POLITÉCNICA DE MADRID
Escuela Técnica Superior de
Ingeniería y Sistemas de Telecomunicación

PROYECTO FIN DE GRADO

APLICACIÓN DE LECTOESCRITURA PARA DISPOSITIVOS CON SISTEMA OPERATIVO ANDROID

JOSÉ CARLOS GARCÍA SECO

Grado en Ingeniería Electrónica de Comunicaciones
Marzo 2015

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA Y SISTEMAS DE TELECOMUNICACIÓN

PROYECTO FIN DE GRADO

TÍTULO: Aplicación de lectoescritura para dispositivos con Sistema Operativo Android

AUTOR: José Carlos García Seco

TITULACIÓN: Grado en Ingeniería Electrónica de Comunicaciones

TUTOR: José Manuel Díaz López

DEPARTAMENTO: Teoría de la Señal y Comunicaciones

VºBº

Miembros del Tribunal Calificador:

PRESIDENTE: Antonio Redondo Hidalgo

VOCAL: José Manuel Díaz López

SECRETARIO: José Luis Rodríguez Vázquez

Fecha de lectura: 18 de marzo de 2015

Calificación:

El Secretario,

Resumen

El presente proyecto pretende ser una herramienta para la enseñanza de la lectoescritura (enseñar a leer y a escribir) para niños con discapacidad, haciendo para ello uso de una aplicación que se ejecuta en una tablet con Sistema Operativo (S.O.) Android. Existe un vacío en el mundo de las aplicaciones para tabletas en este campo en el que se intentará poner un grano de arena para, al menos, tener una aplicación que sirva de toma de contacto a los interesados en este campo.

Para establecer las funcionalidades más adecuadas al propósito de la herramienta, se ha consultado a profesionales de la logopedia de un colegio de educación especial, con cuya colaboración se ha dado forma a la estructura de la misma. La implementación de la aplicación se ha llevado a cabo con programación en entorno Java para Android. Se han incluido diferentes recursos como imágenes, pictogramas y locuciones tanto elementos con licencia libre, como elementos propios generados 'ex profeso' para dar la forma final a la herramienta.

Podemos decir que en general esta aplicación puede ser usada para enseña a leer y escribir a cualquier niño, pero se ha dotado de unas ciertas características que la confieren una orientación especial hacia niños con necesidades educativas especiales. Para ello se ha cuidado mucho la estética, para que ésta sea lo más simple y suave posible, para hacer especial hincapié en la atención de los niños y evitar su distracción con elementos visuales innecesarios. Se ha dotado de estímulos visuales y sonoros para fomentar su interés (aplausos en caso de acierto, colores para diferenciar aciertos y errores, etc.). Se han utilizado los tamaños de letra más grandes posibles (para las discapacidades visuales), etc.

El mercado cuenta con una ingente cantidad de dispositivos Android, con características muy dispares, de tamaño de pantalla, resolución y versiones del S.O. entre otras. La aplicación se ha desarrollado tratando de dar cobertura al mayor porcentaje de ellos posible. El requisito mínimo de tamaño de pantalla sería de siete pulgadas. Esta herramienta no tiene demasiado sentido en dispositivos con pantallas menores por las características intrínsecas de la misma. No obstante se ha trabajado también en la configuración para dispositivos pequeños, como "smartphones", no por su valor como herramienta para la enseñanza de la lectoescritura (aunque en algunos casos podría ser viable) sino más bien con fines de prueba y entrenamiento para profesores, padres o tutores que realizarán la labor docente con dispositivos tablet.

Otro de los requisitos, como se ha mencionado, para poder ejecutar la aplicación sería la versión mínima de S.O., por debajo de la cual (versiones muy obsoletas) la aplicación sería inviable.

Sirva este proyecto pues para cubrir, mediante el uso de la tecnología, un aspecto de la enseñanza con grandes oportunidades de mejora.

Abstract

This Project is aimed to be a tool for teaching reading and writing skills to handicapped children with an Android application. There are no Android applications available on this field, so it is intended to provide at least one option to take contact with.

Speech therapy professionals from a special needs school have been asked for the most suitable functions to be included in this tool. The structure of this tool has been made with the cooperation of these professionals. The implementation of the application has been performed through Java coding for Android. Different resources have been included such as pictures, pictograms and sounds, including free licenses resources and self-developed resources.

In general, it can be said that this application can be used to teach learning and writing skills to any given kid, however it has been provided of certain features that makes it ideal for children with special educational needs. It has been strongly taken into account the whole aesthetic to be as simple and soft as possible, in order to get attention of children, excluding any visual disturbing elements. It has been provided with sound and visual stimulations, to attract their interest (applauses in cases of correct answers, different colours to differentiate right or wrong answers), etc.

There are many different types of Android devices, with very heterogeneous features regarding their screen size, resolution and O.S. version, etc., available today. The application has been developed trying to cover most of them. Minimum screen resolution is seven inches. This tool doesn't seem to be very useful for smaller screens, for its inner features. Nevertheless, it has been developed for smaller devices as well, like smartphones, not intended to be a tool for teaching reading and writing skills (even it could be possible in some cases), but in a test and training context for teachers, parents or guardians who do the teaching work with tablet devices.

Another requirement, as stated before, in order to be able to run the application, it would be the minimum O.S. version, below that (very obsolete versions) the application would become impracticable.

Hope this project to be used to fulfill, by means of technology, one area of teaching with great improvement opportunities.

1. Contenido

2. Lista de acrónimos

S.O.: Sistema Operativo

O.S.: Operating System (en inglés)

TIC: Tecnologías de la Información y las Comunicaciones

SDK: Software Development Kit, kit de desarrollo de software

JDK: Java Development Kit, kit de desarrollo Java

JSR: Java Specification Request

IDE: Integrated Development Environment, entorno de desarrollo integrado

SGBD: Sistema de Gestión de Bases de Datos

ACID: Atomicity, Consistency, Isolation and Durability. Atomicidad, Consistencia, Aislamiento y Durabilidad

WYSIWYG: What You See Is What You Get, lo que ves es lo que obtienes

UML: Unified Modeling Languaje, lenguaje de modelado unificado

GUI: Graphical User Interface, interfaz gráfica de usuario

ETSIST: Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación

PFG: Proyecto de Fin de Grado

ACNEE: Alumnos Con Necesidades Educativas Especiales

3. Introducción

Es evidente que el progreso tecnológico, y más en concreto, el avance en las TIC (Tecnologías de la Información y las Comunicaciones) en los últimos tiempos ha supuesto una verdadera revolución en la forma de desenvolvernos en nuestro día a día.

Los dispositivos móviles han fomentado en gran medida que las TIC estén presente en números aspectos de nuestra vida. Uno de estos aspectos son los nuevos métodos para la formación. Este es el ámbito donde se enmarca el presente proyecto, cuyo objetivo es crear una herramienta orientada a la enseñanza de la lectoescritura, que si bien puede ser usada de forma generalizada, está diseñada haciendo énfasis en las necesidades que presentan los niños con algunos tipos de discapacidad. Se trata de un instrumento educativo que por el momento servirá para complementar los métodos tradiciones de enseñanza, los cuales hacen uso de distintos elementos físicos tradicionales, como tarjetas de cartón con sílabas y pictogramas, entre otros. No obstante se espera que este proyecto sirva como acicate para un incremento en el uso de este tipo de herramientas formativas en dispositivos tan idóneos para este fin como son las tablets.

Existen diferentes tipos de dispositivos tablet en el mercado, como pueden ser las basadas en los sistemas operativos iOS de Apple, Microsoft, o Android. Este proyecto se centrará en el universo Android por tratarse del que cuenta con una mayor cuota de mercado.

Se ha desarrollado una aplicación que dará cobertura a algunas de las actividades habituales en el proceso educativo en los centros de enseñanza para niños con necesidades especiales. Se ha utilizado un entorno de desarrollo típico de programación para dispositivos Android. Para la consecución de los objetivos previstos, se han tenido en cuenta los puntos de vista y opiniones de profesionales de la logopedia, que han aportado su experiencia en el campo de la discapacidad.

La aplicación cuenta con una serie de actividades agrupadas en cinco categorías principales (presentación, discriminación visual, discriminación auditiva, lectura y escritura). La base de trabajo son las sílabas como unidad mínima de enseñanza de la lectoescritura. A través de las diferentes actividades se irán utilizando las sílabas junto con una serie de pictogramas y sonidos relacionados con cada una de dichas sílabas. Se ha tratado de hacer esta aplicación lo más atractiva posible, utilizando elementos de refuerzo que motiven la atención de los niños (aplausos y modificaciones en los colores de elementos visuales cuando se producen aciertos por ejemplo).

En los siguientes apartados se tratará de explicar de forma detallada la planificación y el desarrollo del proyecto así como otra información complementaria que pueden ayudar a comprender mejor el contexto de la misma.

En primer lugar, se ofrecerá una visión general del marco tecnológico aplicado, donde se introducirá el mundo Android y sus entornos de desarrollo. Se hablará también de la base de datos SQLite y de Eclipse.

Después se darán algunas pinceladas sobre la discapacidad, definiciones y clasificaciones. Se tratarán los sistemas aumentativos y alternativos de comunicación.

Posteriormente se detallará como se ha desarrollado todo el proceso del proyecto, tratando todos los puntos importantes, como los antecedentes, el entorno de desarrollo, las fases por las que se

ha pasado, una descripción completa de la aplicación y su implementación y algunas situaciones singulares que ha sido necesario abordar para el correcto desempeño de la misma.

A continuación se presentan las conclusiones, la bibliografía y referencias bibliográficas.

Por último se desarrollan cuatro anexos, el primero con información sobre sílabas, el segundo sobre los pictogramas utilizados, el tercero sobre el emulador Genymotion, y el cuarto con el manual de instrucciones de la aplicación.

4. Marco tecnológico

4.1 Introducción a Android

Android es una plataforma para dispositivos móviles que presenta una serie de características que lo hacen diferente al resto (iOS, Symbian, Windows Phone, BlackBerry, Palm, Java Mobile Edition, Linux Mobile (LiMo), etc.). Es la primera que combina en una misma solución las siguientes cualidades:

- **Plataforma realmente abierta.** Es una plataforma de desarrollo libre basada en Linux y de código abierto. Una de sus grandes ventajas es que se puede usar y customizar el sistema sin pagar royalties.

- **Adaptable a cualquier tipo de hardware**. Android no ha sido diseñado exclusivamente para su uso en teléfonos y tabletas. Hoy en día podemos encontrar relojes, cámaras, electrodomésticos y gran variedad de sistemas empotrados que se basan en este sistema operativo. Este hecho tiene sus evidentes ventajas, pero también va a suponer un esfuerzo adicional al programador. La aplicación ha de funcionar correctamente en dispositivos con gran variedad de tipos de entrada, pantalla, memoria, etc. Esta característica contrasta con la estrategia de Apple. En iOS tenemos que desarrollar una aplicación para iPhone y otra diferente para iPad.

- **Portabilidad asegurada**. Las aplicaciones finales son desarrolladas en Java lo que nos asegura que podrán ser ejecutadas en cualquier tipo de CPU, tanto presente como futuro. Esto se consigue gracias al concepto de máquina virtual.

- **Arquitectura basada en componentes inspirados en Internet**. Por ejemplo, el diseño de la interfaz de usuario se hace en xml, lo que permite que una misma aplicación se ejecute en un móvil de pantalla reducida o en un TV.

- **Filosofía de dispositivo siempre conectado a Internet**.

- **Gran cantidad de servicios incorporados**. por ejemplo, localización basada tanto en GPS como en redes, bases de datos con SQL, reconocimiento y síntesis de voz, navegador, multimedia.

- **Aceptable nivel de seguridad**. Los programas se encuentran aislados unos de otros gracias al concepto de ejecución dentro de una caja que hereda de Linux. Además, cada aplicación dispone de una serie de permisos que limitan su rango de actuación (servicios de localización, acceso a Internet, etc.)

- **Optimizado para baja potencia y poca memoria**. Por ejemplo, Android utiliza la Máquina Virtual Dalvik. Se trata de una implementación de Google de la máquina virtual de Java optimizada para dispositivos móviles.

- **Alta calidad de gráficos y sonido**. gráficos vectoriales suavizados, animaciones inspiradas en Flash, gráficos en 3 dimensiones basados en OpenGL. Incorpora codecs estándar más comunes de audio y vídeo, incluyendo H.264 (AVC), MP3, AAC, etc.

Las aplicaciones desarrolladas en Android se escriben en el lenguaje de programación Java, y en el de intercambio XML (para la construcción de layouts o interfaces gráficas). El código, por lo tanto, se desarrolla en un entorno de programación para Java, como puede ser: NetBeans o Eclipse, y con las herramientas de Android instaladas, se compila el código y se genera un fichero "apk". Será este fichero de tipo apk el que se instale en el dispositivo móvil.

Android facilita todas las interfaces necesarias para acceder a las funciones del teléfono, como por ejemplo el GPS del móvil, las llamadas de teléfono, la agenda, etc. Así se facilita la entrada de más aplicaciones desarrolladas por diferentes programadores y con costes muy bajos. Cualquier persona puede bajarse el código fuente, lo puede inspeccionar, trabajar, cambiar etc. Esto le da seguridad al usuario, ya que es un código abierto que le permite detectar fallos más rápidamente y así repararlos.

El S.O. Android es un Sistema Operativo de tipo Multiusuario, basado en Linux.

Android presenta algunas características interesantes:

- **Framework de aplicaciones.** permite el reemplazo y la reutilización de los componentes.
- **Navegador integrado.** Basado en el motor open Source Webkit.
- **SQLite.** Base de datos para almacenamiento estructurado que se integra directamente con las aplicaciones.
- **Multimedia.** Soporte para medios con formatos comunes de audio, video e imágenes planas (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
- **Máquina virtual Dalvik.** Base de llamadas de instancias muy similar a Java.
- **Telefonía GSM.** Dependiente del terminal.
- **Bluetooth, EDGE, 3g y Wifi.** Dependiente del terminal.
- **Cámara GPS, brújula, y acelerómetro.** Dependiente del terminal.
- **Pantalla táctil**

Podemos decir que Android nos ofrece una forma sencilla y novedosa de implementar potentes aplicaciones para diferentes tipos de dispositivo.

4.2 Los orígenes

Google adquiere Android Inc. en el año 2005. Se trataba de una pequeña compañía, que acababa de ser creada, orientada a la producción de aplicaciones para terminales móviles. Ese mismo año empiezan a trabajar en la creación de una máquina virtual Java optimizada para móviles (Dalvik VM).

En el año 2007 se crea el consorcio Handset Alliance [1] con el objetivo de desarrollar estándares abiertos para móviles. Está formado por Google, Intel, Texas Instruments, Motorola, T-Mobile, Samsung, Ericsson, Toshiba, Vodafone, NTT DoCoMo, Sprint Nextel y otros. Una pieza clave de los objetivos de esta alianza es promover el diseño y difusión de la plataforma

Android. Sus miembros se han comprometido a publicar una parte importante de su propiedad intelectual como código abierto bajo licencia Apache v2.0.

En noviembre del 2007 se lanza una primera versión del Android SDK (Software Development Kit o kit de desarrollo de software). Al año siguiente aparece el primer móvil con Android (T-Mobile G1). En octubre Google libera el código fuente de Android principalmente bajo licencia de código abierto Apache (licencia GPL v2 para el núcleo). Ese mismo mes se abre Android Market, para la descarga de aplicaciones. En abril del 2009 Google lanza la versión 1.5 del SDK que incorpora nuevas característica como el teclado en pantalla. A finales del 2009 se lanza la versión 2.0 y durante el 2010 las versiones 2.1, 2.2 y 2.3.

Durante el año 2010 Android se consolida como uno de los sistemas operativos para móviles más utilizados, con resultados cercanos a iOS e incluso superando al sistema de Apple en EE.UU.

En el 2011 se lanzan la versión 3.0, 3.1 y 3.2 específica para tabletas y la 4.0 tanto para móviles como para tabletas. Durante este año Android se consolida como la plataforma para móviles más importante alcanzando una cuota de mercado superior al 50%.

En 2012 Google cambia su estrategia en su tienda de descargas online, reemplazando Android Market por Google Play Store. Donde en un solo portal unifica tanto la descarga de aplicaciones como de contenidos. En este año aparecen las versiones 4.1 y 4.2 del SDK. Android mantiene su espectacular crecimiento, alcanzando a finales de año una cuota de mercado del 70%. En 2013 se lanzan las versiones 4.3 y 4.4 (KitKat). A finales de 2014 se lanza la versión 5.0 (Lollipop) y la cuota de mercado de teléfonos inteligentes Android alcanza en el tercer cuatrimestre de ese mismo año casi el 85% [2].

4.3 Versiones de Android

4.3.1 Android Beta

La versión beta de Android fue lanzada el 5 de noviembre de 2007, mientras el Software Development kit (SDK) fue lanzado el 12 de noviembre de 2007. Las versiones públicas Beta del SDK fueron lanzados en el siguiente orden:

- 16 de noviembre de 2007: m3-rc22a
- 14 de diciembre de 2007: m3-rc37a
- 13 de febrero de 2008: m5-rc14
- 3 de marzo de 2008: m5-rc15
- 18 de agosto de 2008: 0.9
- 23 de septiembre de 2008: 1.0-r1

4.3.2 Android 1.0 Apple Pie

Android 1.0 Apple Pie (Tarta de manzana), la primera versión comercial del software, fue lanzado el 23 septiembre de 2008.11 El primer dispositivo Android fue el HTC Dream.

4.3.3 Android 1.1 Banana Bread

El 9 de febrero de 2009, La actualización Android 1.1 Banana Bread (Pan de plátano) fue lanzada, inicialmente solo para el HTC Dream así que solo sirve para este teléfono. Android 1.1

fue conocido como "Petit Four" internamente, aunque este nombre no se utilizó oficialmente. La actualización resolvió fallos, cambió la API y agregó una serie de características nuevas.

4.3.4 Android 1.5 Cupcake

El 30 de abril de 2009, La actualización de Android 1.5 Cupcake fue lanzada, basada en núcleo Linux 2.6.27. La actualización incluye varias nuevas características y correcciones de interfaz de usuario.

4.3.5 Android 1.6 Donut

El 15 de septiembre de 2009, fue lanzado el SDK de Android 1.6 Donut, basado en el núcleo Linux 2.6.29. En la actualización se incluyen numerosas características nuevas.

4.3.6 Android 2.0/2.1 Eclair

El 26 de octubre de 2009, el SDK de Android 2.0 – con nombre en clave Eclair – fue lanzado, basado en el núcleo de Linux 2.6.29.

4.3.7 Android 2.2.x Froyo

El 20 de mayo de 2010, El SDK de Android 2.2 Froyo (Yogur helado) fue lanzado, basado en el núcleo Linux 2.6.32

4.3.8 Android 2.3.x Gingerbread

El 6 de diciembre de 2010, el SDK de Android 2.3 Gingerbread (Pan de Jengibre) fue lanzado, basado en el núcleo Linux 2.6.35

4.3.9 Android 3.x Honeycomb

El 22 de febrero de 2011, sale el SDK de Android 3.0 Honeycomb (Panal de Miel). Fue la primera actualización exclusiva para tablet, lo que quiere decir que sólo es apta para tablets y no para teléfonos Android. Está basada en el núcleo de Linux 2.6.36. El primer dispositivo con esta versión fue la tableta Motorola Xoom, lanzado el 24 de febrero de 2011

4.3.10 Android 4.0.x Ice Cream Sandwich

El SDK para Android 4.0.0 Ice Cream Sandwich (Sandwich de Helado), basado en el núcleo de Linux 3.0.1, fue lanzado públicamente el 19 de octubre de 2011. Gabe Cohen de Google declaró que Android 4.0 era "teóricamente compatible" con cualquier dispositivo Android 2.3 en producción en ese momento, pero sólo si su procesador y memoria RAM lo soportaban. El código fuente para Android 4.0 se puso a disposición el 14 de noviembre de 2011. La actualización incluye numerosas novedades.

4.3.11 Android 4.1 Jelly Bean.

Google anunció Android 4.1 Jelly Bean (Gomita Confitada o Gominola) en conferencia el 27 de junio de 2012. Basado en el núcleo de Linux 3.0.31, Bean fue una actualización incremental con el enfoque primario de mejorar la funcionalidad y el rendimiento de la interfaz de usuario. La mejora de rendimiento involucró el "Proyecto Butter", el cual usa anticipación táctil, triple buffer, latencia vsync extendida y un arreglo en la velocidad de cuadros de 60 fps para crear una fluida y "mantecosa"-suavidad de la interfaz de usuario. Android 4.1 Jelly Bean fue lanzado bajo AOSP el 9 de julio de 2012, y el Nexus 7, el primer dispositivo en correr Jelly Bean, fue lanzado el 13 de julio de 2012

4.3.12 Android 4.2 Jelly Bean (Gummy Bear)

Se esperaba que Google anunciara Jelly Bean 4.2 en un evento en la ciudad de Nueva York el 29 de octubre de 2012, pero el evento fue cancelado debido al Huracán Sandy. En lugar de

reprogramar el evento en vivo, Google anunció la nueva versión con un comunicado de prensa, bajo el eslogan "A new flavor of Jelly Bean". El primer dispositivo en correr Android 4.2 fue el Nexus 4 de LG y el Nexus 10 de Samsung, los cuales fueron lanzados el 13 de noviembre de 2012

4.3.13 Android 4.3 Jelly Bean

Google lanzó Jelly Bean 4.3, bajo el lema "Una forma aún más dulce Jelly Bean" el 24 de julio de 2013, durante un evento en San Francisco llamado "Desayuno con Sundar Pichai". La versión hizo su debut en la nueva generación de segundo Nexus 7 que se estrenó el 30 de julio del 2013

4.3.14 Android 4.4 KitKat

Su nombre se debe a la chocolatina KitKat, de la empresa internacional Nestlé. Cuenta con una enorme cantidad de nuevas características respecto a su versión anterior.

4.3.15 Android 5.0 Lollipop

Android 5.0 Lollipop es la última versión del sistema operativo para dispositivos móviles Android en el momento de la redacción de este libro. Fue dada a conocer el 25 de junio de 2014 durante el Google I/O 2014 como Android L y su versión beta fue liberada el día siguiente a determinados dispositivos Google Nexus, concretamente el Nexus 5 y la tablet Nexus 7 2013. El 15 de octubre de 2014, Google dio el nombre oficial de Android L: Lollipop (al español: piruleta o paletita) y la versión es 5.0, siendo anunciado oficialmente junto al Nexus 6, Nexus 9 y Nexus Player

4.4 Arquitectura Android

Antes de hablar de la arquitectura de la plataforma Android, debemos definir lo que es una plataforma en términos de tecnología informática. En informática, una plataforma es un sistema que sirve como base para hacer funcionar determinados módulos de hardware o de software con los que es compatible. Dicho sistema está definido por un estándar alrededor del cual se determina una arquitectura de hardware y una plataforma de software (incluyendo entornos de aplicaciones). Al definir plataformas se establecen los tipos de arquitectura, sistema operativo, lenguaje de programación o interfaz de usuario compatibles [3].

La plataforma Android presenta la arquitectura que podemos ver en la figura 1.

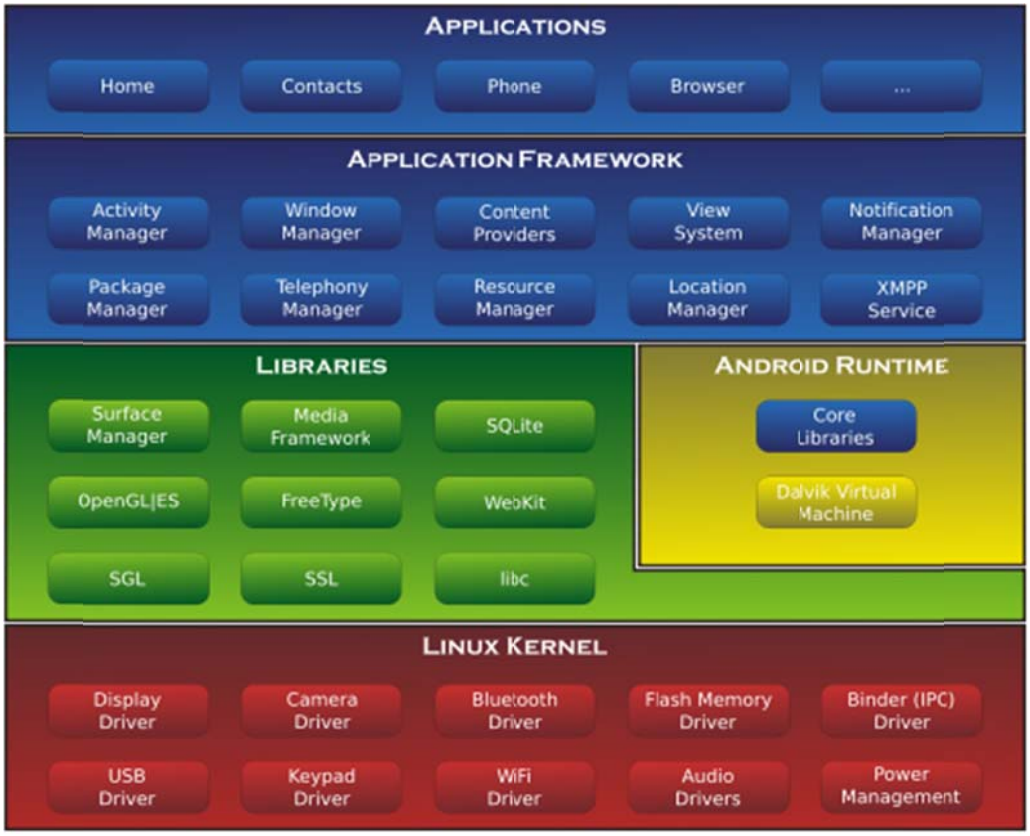


Figura 1. Arquitectura Plataforma Android

Haremos un breve repaso sobre los elementos de esta arquitectura:

- Aplicaciones (Applications)

Las aplicaciones creadas con la plataforma Android, incluirán como base un cliente de e-mail, calendario, programa de SMS, mapas, navegador, contactos, y algunos otros servicios mínimos. Todo ello escrito en el lenguaje de programación Java.

- Framework de aplicaciones (Application Framework)

El código fuente usado en las aplicaciones base es accesible para los desarrolladores de aplicaciones. De esta forma no se generan cientos de componentes de aplicaciones distintas, que respondan a la misma acción, dando la posibilidad de que los programas sean modificados o reemplazados por cualquier usuario sin tener que empezar a programar sus aplicaciones desde el principio.

- Librerías

Android incluye en su base de datos un juego de librerías C/C++, que son expuestas a todos los desarrolladores a través del framework de las aplicaciones Android System C library, librerías de medios, librerías de gráficos, 3D, SQLite, manejo de pantalla, mapas de bits y tipos de letra, etc.

Junto a estas librerías, encontramos lo necesario para la ejecución de las aplicaciones a través de Dalvik (máquina virtual). Cada aplicación utiliza una instancia de la máquina virtual ejecutando

un archivo DEX (Dalvik Executable) y el sistema está optimizado para que se ejecuten múltiples instancias de la máquina virtual.

- Runtime de Android

Android incorpora un juego de librerías que aportan la mayor parte de las funcionalidades disponibles en las librerías base del lenguaje de programación Java. La Máquina Virtual está basada en registros, y ejecuta clases compiladas por el compilador de Java que anteriormente han sido transformadas al formato .dex (Dalvik Executable) por la herramienta "dx".

La máquina virtual de Dalvik se basa en el kernel de Linux para la funcionalidad subyacente como subprocesos y de bajo nivel de gestión de memoria

- Kernel de Linux

Android se basa en el kernel (núcleo) de Linux que actúa como una capa de abstracción entre el hardware y el resto del conjunto de software, además de prestar los servicios de seguridad, gestión de memoria, gestión de procesos, network stack, driver model.

Aunque Android utiliza como base el kernel de Linux, los dos sistemas no son lo mismo. Android no cuenta con un sistema nativo de ventanas de Linux ni tiene soporte para glibc (librería estándar de C), ni tampoco es posible utilizar la mayoría de aplicaciones de GNU de Linux.

Además de todo lo implementado en el kernel de Linux, Android agrega elementos específicos para plataformas móviles como la comunicación entre procesos (lograda a través del binder), la forma de manejar la memoria compartida (ashmem) y la administración de energía (con wakelocks).

4.5 La máquina virtual Dalvik

En Android, las aplicaciones se programan en el lenguaje Java y se ejecutan mediante esta máquina virtual (Dalvik), específicamente diseñada para Android. Esta máquina virtual ha sido optimizada y adaptada a las peculiaridades propias de los dispositivos móviles (menor capacidad de proceso, baja memoria, alimentación por batería, etc.) y trabaja con ficheros de extensión .dex (Dalvik Executables). Dalvik no trabaja directamente con el bytecode de Java, sino que lo transforma en un código más eficiente que el original orientado a procesadores pequeños. Gracias a la herramienta "dx", esta transformación es posible: los ficheros .class de Java se compilan en ficheros .dex, de forma que cada fichero .dex puede contener varias clases. Después, este resultado se comprime en un único archivo de extensión .apk (Android Package), que es el que se distribuirá en el dispositivo móvil.

Dalvik permite varias instancias simultáneas de la máquina virtual, y a diferencia de otras máquinas virtuales, está basada en registros y no en pila, lo que implica que las instrucciones son más reducidas y el número de accesos a memoria es menor.

La utilización de esta máquina virtual responde a un deseo de mejorar y optimizar la ejecución de aplicaciones en dispositivos móviles, así como evitar la fragmentación de otras plataformas como Java ME (Java Micro Edition). A pesar de esta aclaración oficial, no ha pasado inadvertido que con esta maniobra Android ha esquivado tener que utilizar directamente Java

ME, evitando así que los futuros desarrolladores de aplicaciones tengan que comprar alguna licencia, y tampoco publicar el código fuente de sus productos.

El lenguaje Java utilizado en Android no sigue ningún JSR (Java Specification Request) determinado, y Google se afana en recordar que Android no es tecnología Java, sino que simplemente utiliza este lenguaje en sus aplicaciones.

4.6 Componentes básicos de una aplicación Android

Una vez vista la arquitectura, empezaremos con lo fundamental para desarrollar una aplicación. Los componentes básicos de una aplicación son los siguientes:

Vista (view), layout, actividad (activity), servicio (service), intención (intent), fragment, receptor de anuncios (broadcast receiver), proveedor de contenidos (content provider). Haremos una breve descripción de los componentes principales en esta aplicación:

- **Vistas.** Son los elementos que componen la interfaz de usuario de una aplicación: por ejemplo, un botón o una entrada de texto. Todas las vistas van a ser objetos descendientes de la clase View, y por tanto, pueden ser definidas utilizando código Java. Sin embargo, lo habitual será definir las vistas utilizando un fichero XML y dejar que el sistema cree los objetos por nosotros a partir de este fichero. Esta forma de trabajar es muy similar a la definición de una página web utilizando código HTML.

- **Layout.** Es un conjunto de vistas agrupadas de una forma determinada. Vamos a disponer de diferentes tipos de layouts para organizar las vistas de forma lineal, en cuadrícula o indicando la posición absoluta de cada vista. Los layouts también son objetos descendientes de la clase View. Igual que las vistas, los layouts pueden ser definidos en código, aunque la forma habitual de definirlos es utilizando código XML.

- **Actividad.** Una aplicación en Android va a estar formada por un conjunto de elementos básicos de visualización, coloquialmente conocidos como pantallas de la aplicación. En Android cada uno de estos elementos, o pantallas, se conoce como actividad. Su función principal es la creación del interfaz de usuario. Una aplicación puede necesitar varias actividades para crear el interfaz de usuario. Las diferentes actividades creadas serán independientes entre sí, aunque todas trabajarán para un objetivo común. Toda actividad ha de pertenecer a una clase descendiente de Activity.

- **Intent.** Los intents nos proporcionan un medio de pasar datos entre actividades cuando queremos lanzar una nueva actividad y en general para el intercambio de información entre componentes.

4.7 Ciclo de vida de una actividad

En Android, cada aplicación se ejecuta en su propio proceso, aportando beneficios en cuestiones básicas como seguridad, gestión de memoria, o la ocupación de la CPU del dispositivo móvil. Android también se ocupa de lanzar y parar todos estos procesos, gestionar su ejecución y decidir qué hacer en función de los recursos disponibles y de las órdenes dadas por el usuario.

El usuario desconoce esta forma de trabajar de Android, simplemente es consciente de que mediante un simple clic pasa de una a otra aplicación y puede volver a cualquiera de ellas en el momento que lo desee. El usuario no se preocupa de cuál es la aplicación que realmente está activa, cuánta memoria está consumiendo, ni si existen o no recursos suficientes para abrir una aplicación adicional. Todo eso son tareas propias del sistema operativo.

Android lanza tantos procesos como permitan los recursos del dispositivo. Cada proceso correspondiente a una aplicación, estará formado por una o varias actividades independientes (activities) de esa aplicación. Cuando el usuario navega de una actividad a otra, o abre una nueva aplicación, el sistema duerme dicho proceso y realiza una copia de su estado para poder recuperarlo más tarde. El proceso y la actividad siguen existiendo en el sistema, pero están dormidos y su estado ha sido guardado. Es entonces cuando crea, o despierta (si ya existe) el proceso para la aplicación que debe ser lanzada asumiendo que existan recursos para ello.

Cada uno de los componentes básicos de Android tiene un ciclo de vida bien definido; esto implica que el desarrollador puede controlar en cada momento en qué estado se encuentra dicho componente, pudiendo así programar las acciones que mejor convengan. El componente Activity, probablemente el más importante, tiene un ciclo de vida como el mostrado en la figura 2.

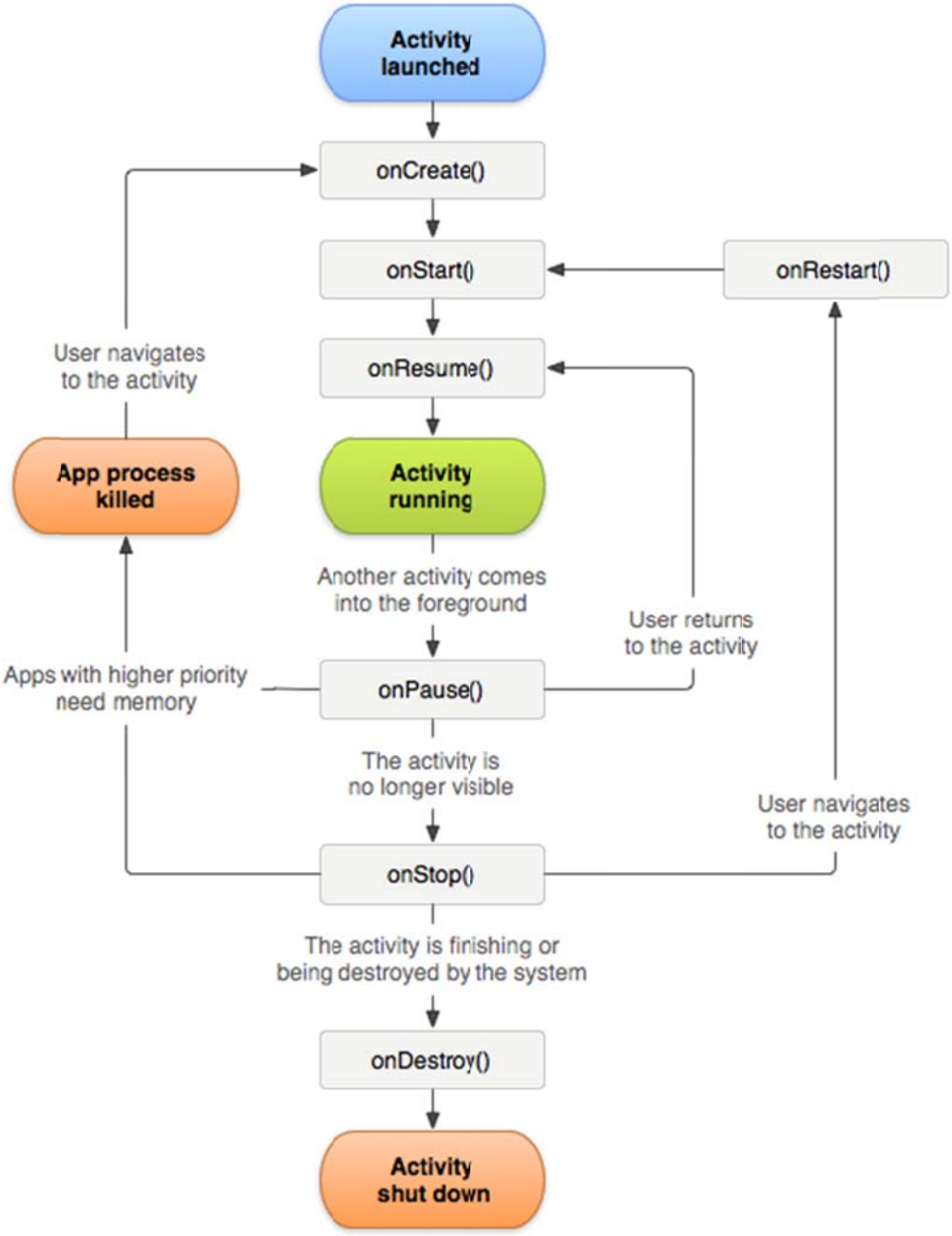


Figura 2 Ciclo de vida de una actividad

- onCreate(), onDestroy(): Abarcan todo el ciclo de vida. Cada uno de estos métodos representan el principio y el fin de la actividad.

- onStart(), onStop(): Representan la parte visible del ciclo de vida. Desde onStart() hasta onStop(), la actividad será visible para el usuario, aunque es posible que no tenga el foco de acción por existir otras actividades superpuestas con las que el usuario está interactuando. Pueden ser llamados múltiples veces.

- onResume(), onPause(): Delimitan la parte útil del ciclo de vida. Desde onResume() hasta onPause(), la actividad no sólo es visible, sino que además tiene el foco de la acción y el usuario puede interactuar con ella.

Tal y como se ve en la figura 2, el proceso que mantiene a la Activity puede ser eliminado cuando se encuentra en onPause() o en onStop(), es decir, cuando no tiene el foco de la aplicación. Android nunca elimina procesos con los que el usuario está interactuando en ese momento. Una vez se elimina el proceso, el usuario desconoce dicha situación y puede incluso volver atrás y querer usarlo de nuevo. Entonces el proceso se restaura gracias a una copia y vuelve a estar activo como si no hubiera sido eliminado. Además, la Activity puede haber estado en segundo plano, invisible, y entonces es despertada pasando por el estado onRestart().

¿Pero qué ocurre en realidad cuando no existen recursos suficientes?. Lógicamente, los recursos son limitados, más aun cuando se trata de dispositivos móviles. En el momento en el que Android detecta que no hay los recursos necesarios para poder lanzar una nueva aplicación, analiza los procesos existentes en ese momento y elimina los procesos que sean menos prioritarios para poder liberar sus recursos.

Cuando el usuario regresa a una actividad que está dormida, el sistema simplemente la despierta. En este caso, no es necesario recuperar el estado guardado porque el proceso todavía existe y mantiene el mismo estado. Sin embargo, cuando el usuario quiere regresar a una aplicación cuyo proceso ya no existe porque se necesitaba liberar sus recursos, Android lo crea de nuevo y utiliza el estado previamente guardado para poder restaurar una copia fresca del mismo. Como se ha explicado anteriormente, el usuario no percibe esta situación ni conoce si el proceso ha sido eliminado o está dormido.

4.8 Seguridad en Android

En Android, como ya se ha indicado anteriormente cada aplicación se ejecuta en su propio proceso. La mayoría de las medidas de seguridad entre el sistema y las aplicaciones deriva de los estándares de Linux, cuyo kernel constituye el núcleo principal de Android.

Cada proceso en Android constituye lo que se llama un cajón de arena o sandbox, que proporciona un entorno seguro de ejecución. Por defecto, ninguna aplicación tiene permiso para realizar ninguna operación o comportamiento que pueda impactar negativamente en la ejecución de otras aplicaciones o del sistema mismo. Por ejemplo, acciones como leer o escribir ficheros privados del usuario (contactos, teléfonos, etc.), leer o escribir ficheros de otras aplicaciones, acceso de red, habilitación de algún recurso hardware del dispositivo, etc., no están permitidas. La única forma de poder saltar estas restricciones impuestas por Android, es mediante la declaración explícita de un permiso que autorice a llevar a cabo una determinada acción habitualmente prohibida.

Además, en Android es obligatorio que cada aplicación esté firmada digitalmente mediante un certificado, cuya clave privada sea la del desarrollador de dicha aplicación.

No es necesario vincular a una autoridad de certificado, el único cometido del certificado es crear una relación de confianza entre las aplicaciones. Mediante la firma, la aplicación lleva adjunta su autoría.

Para establecer un permiso para una aplicación, es necesario declarar en el manifiesto uno o más elementos <uses-permission> donde se especifica el tipo de permiso que se desea habilitar.

4.9 Interfaces de usuario (View, Layouts)

La interfaz de usuario se define en los archivos XML del directorio res/layout. Cada pantalla tendrá un código XML diferente.

Diseñar una pantalla usando Java puede resultar complejo y poco eficiente, sin embargo, Android soporta XML para diseñar pantallas y define elementos personalizados, cada uno representando a una "subclase" especifica de View. Se pueden crear pantallas de la misma manera que se diseñan ficheros HTML.

Cada fichero describe un layout (una pantalla) y cada layout a su vez puede contener otros elementos. Para gestionar la interfaz de usuario, Android introduce las siguientes terminologías:

- **View.** Una View es un objeto cuya clase es android.view.View. Es una estructura de datos cuyas propiedades contienen los datos de la capa, la información específica del área rectangular de la pantalla y permite establecer el layout. Una View tiene: layout, drawing, focus change, scrolling, etc.
  La clase view es útil como clase base para los widgets, que son unas subclases ya implementadas que dibujan los elementos en la pantalla. Los widgets contienen sus propias medidas, pero puedes usarlas para construir tu interfaz más rápidamente. La lista de widgets que puedes utilizar incluye los siguientes: Text, EditText, InputMethod, MovementMethod, Button, RadioButton, CheckBox, y ScrollView.

- **Viewgroups.** Un viewgroup es un objeto de la clase android.view.Viewgroup, como su propio nombre indica, un viewgroup es un objeto especial de view cuya función es contener y controlar la lista de views y de otros viewgroups.
  Los viewgroups te permiten añadir estructuras a la interfaz y acumular complejos elementos en la pantalla que son diseccionados por una sola entidad. La clase viewgroup es útil como base de la clase layouts, que son subclases implementadas que proveen los tipos más comunes de los layouts de pantalla. Los layouts proporcionan una manera de construir una estructura para una lista de views.

- **Árbol estructurado de la interfaz UI.** En la plataforma Android se define una Activity del UI usando un árbol de nodos view y viewgroups, como vemos en la imagen de abajo. El árbol puede ser tan simple o complejo como necesites hacerlo, y se puede desarrollar usando los widgets y layouts que Android proporciona o creando tus propios views. Los views y viewgroups deben estar contenidos en los layouts, los cuales contienen otros elementos presentes en una vista. Dentro de cada layout podemos poner todos los elementos necesarios, incluidos otros layouts. Así conseguiremos estructurar la pantalla de la manera deseada. Existen una gran variedad de layouts, en función de su posicionamiento en la pantalla y se describen a continuación los más importantes:

- **LinearLayout.** Se les conoce como contenedores y sirven para reorganizar los elementos de nuestra aplicación. Sus hijos son los views, viewgroup y/o otros layouts. Nos permitirán alinear sus hijos en una única dirección, ya sea horizontal o vertical

como se muestra en las siguientes imágenes. La orientación predeterminada es horizontal.

- **RelativeLayout.** En este caso todos los elementos van colocados en una posición relativa a otro. Aquí podemos jugar con las distancias entre elementos en la pantalla, la cual se expresa en pixeles.

- **Absolute layout.** Coloca los elementos en posiciones absolutas en la pantalla, teniendo en cuenta que la posición (0,0) es el extremo superior izquierdo de la pantalla.

- **TableLayout.** Permite colocar los elementos en forma de tabla. Se utiliza el elemento <TableRow> para designar a una fila de la tabla. Cada fila puede tener uno o más puntos de vista. Cada vista se coloca dentro de una fila en forma de celda.

- **FrameLayout.** Es un marcador de posición que puede usarse para mostrar una única vista. Se pueden agregar múltiples puntos de vista a un FrameLayout pero cada uno se acumulará en la parte superior izquierda de la anterior.

4.10 Instalación y requisitos del SDK de Android

Los siguientes pasos son una adaptación de la guía de instalación para desarrollo de http://developer.android.com/.

Para poder empezar a desarrollar aplicaciones en Android, primero debemos asegurarnos que tenemos correctamente instalado JDK (Java Development Kit, kit de desarrollo Java), se puede descargar desde la web de Oracle http://www.oracle.com/technetwork/es/java/javase/downloads/index.html

Es necesario descargar e instalar el SDK de Android. La descarga se puede hacer desde http://developer.android.com/sdk/index.html.

Posteriormente es recomendable instalar un IDE (Integrated Development Environment, entorno de desarrollo integrado) para desarrollar las aplicaciones. Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación; es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDE pueden ser aplicaciones por sí solas o pueden ser parte de aplicaciones existentes. Proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación. El IDE por excelencia en la programación Android ha sido Eclipse https://eclipse.org/, aunque últimamente está tomando fuerza Android Studio http://developer.android.com/sdk/index.html

4.11 Emuladores Android

Para hacer pruebas de una aplicación en fase de desarrollo, podemos hacer uso de un dispositivo móvil real, como tablets o smartphones, o bien, en numerosas ocasiones es más conveniente usar emuladores que corren en el mismo ordenador donde se está desarrollando el código. El emulador de Android se comporta como si fuera un dispositivo normal, se puede simular llamadas, enviar SMS, enviar señales GPS, navegar por internet, utilizar Google Maps, etc.

Los emuladores proporcionan una ventaja insuperable sobre los dispositivos físicos y es que podemos crear un entorno de pruebas a medida, es decir, podemos seleccionar la versión de Android donde ejecutar la aplicación, así como el tamaño y resolución de pantalla, configuración de procesador, memoria, almacenamiento, y otros parámetros del sistema. Contar con un entorno de pruebas heterogéneo es imprescindible para hacer un testeo completo cuando sea necesario. La desventaja es que el rendimiento del sistema emulado suele ser inferior al de un dispositivo real.

Eclipse cuenta con plugins de emulador, aunque el rendimiento es realmente pobre. Genymotion es una aplicación que hace uso del entorno de máquinas virtuales VirtualBox y que se integra perfectamente con Eclipse. Su rendimiento mejora bastante con respecto al anteriormente citado. Se hablará un poco más detenidamente de Genymotion en el Anexo III.

Existen otros emuladores, como el que presenta Android Studio integrado en el propio entorno de desarrollo, pero se ha elegido Genymotion como la opción más adecuada para completar el entorno de desarrollo.

4.12 Crear un proyecto en Android

A continuación se detalla el proceso de creación de un proyecto de Android usando el IDE utilizado para desarrollar la aplicación que ocupa este Proyecto Fin de Grado (PFG), que no es otro que Eclipse.

Abrimos Eclipse y pinchamos en la barra de menú sobre la opción File > New > Android Application Project (figura 3).

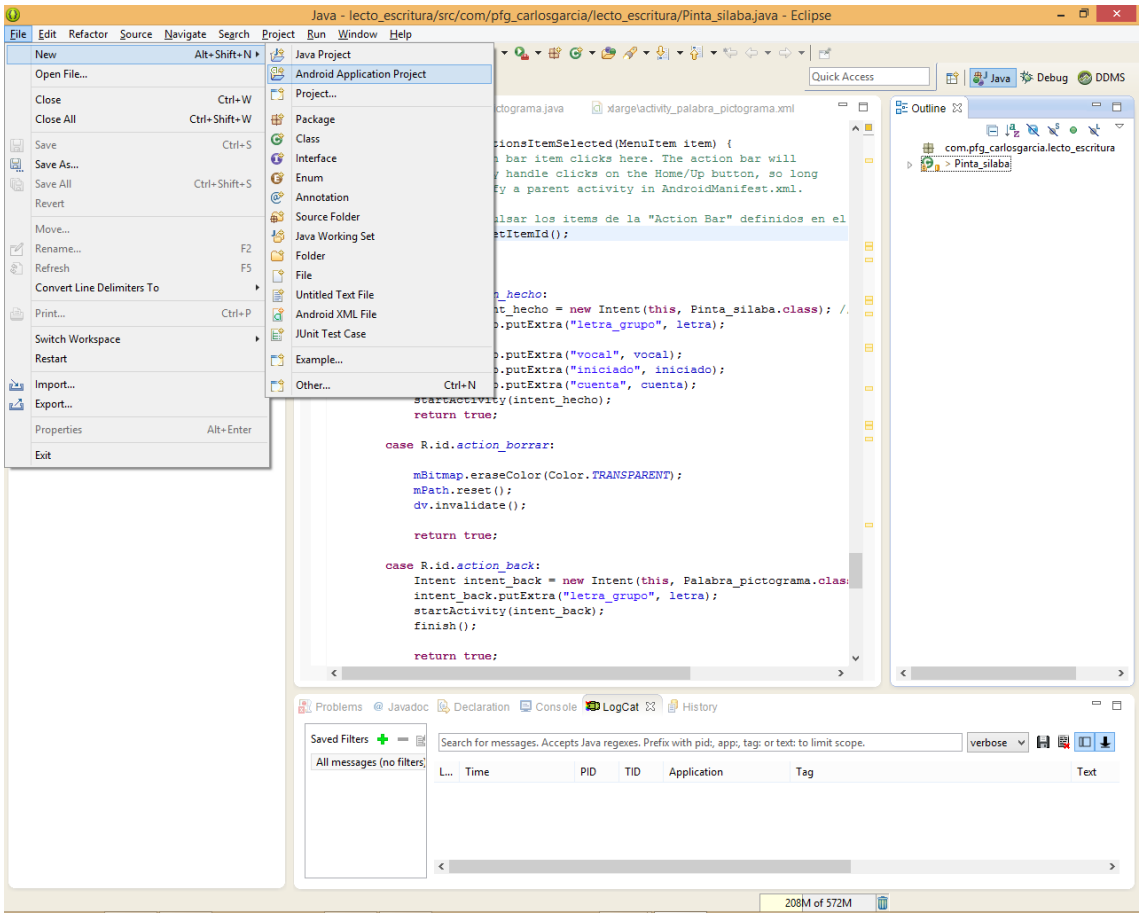


Figura 3. Creación del proyecto Android con Eclipse

A continuación rellenamos los datos básicos del proyecto: nombre de aplicación, nombre de proyecto, nombre de paquete, versiones mínima y máxima de Android, versión de compilación y tema de la aplicación (figura 4). Una vez terminado, pinchamos el botón Next.

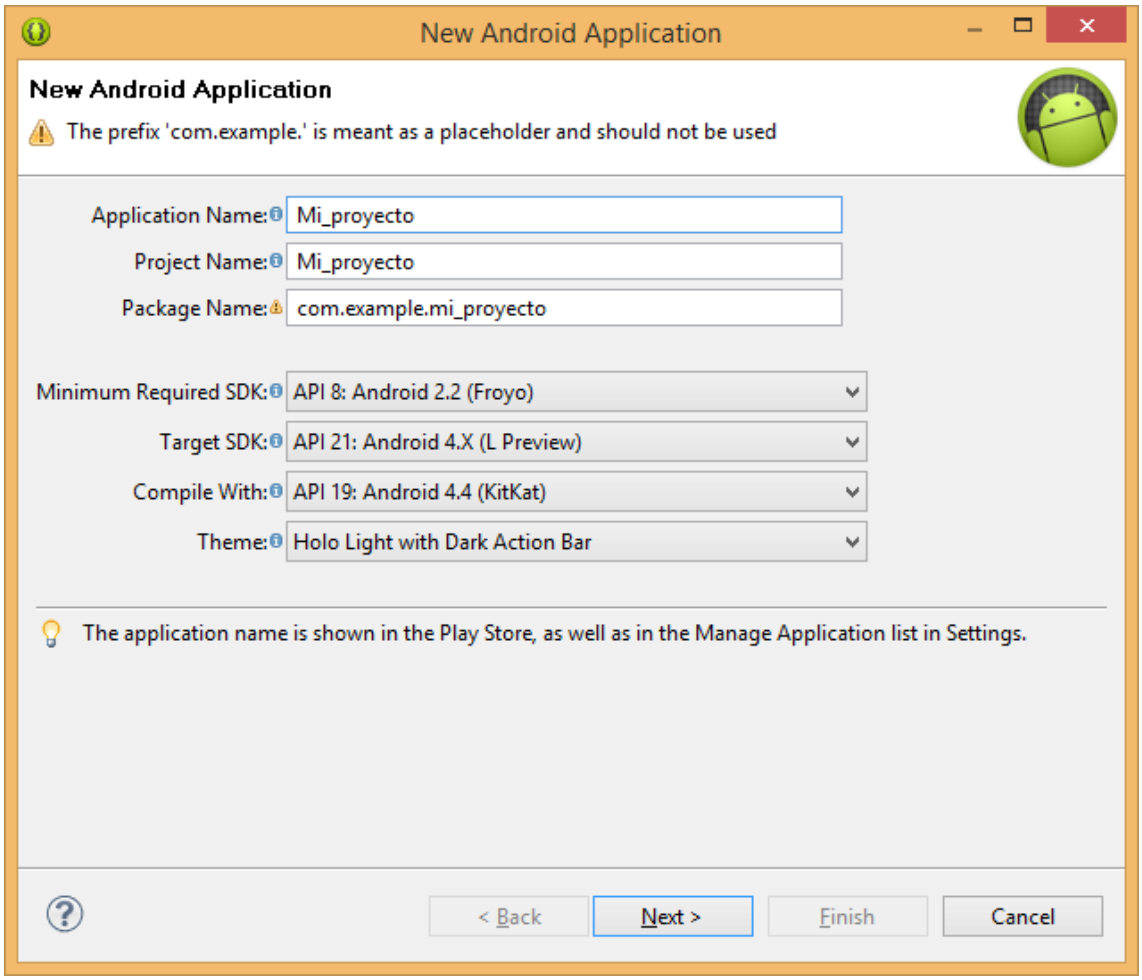


Figura 4. Datos del proyecto.

Definimos ahora otras configuraciones, donde es importante asignar cual será el directorio de trabajo para el proyecto Workspace o WS (figura 5). Pinchamos el botón Next.

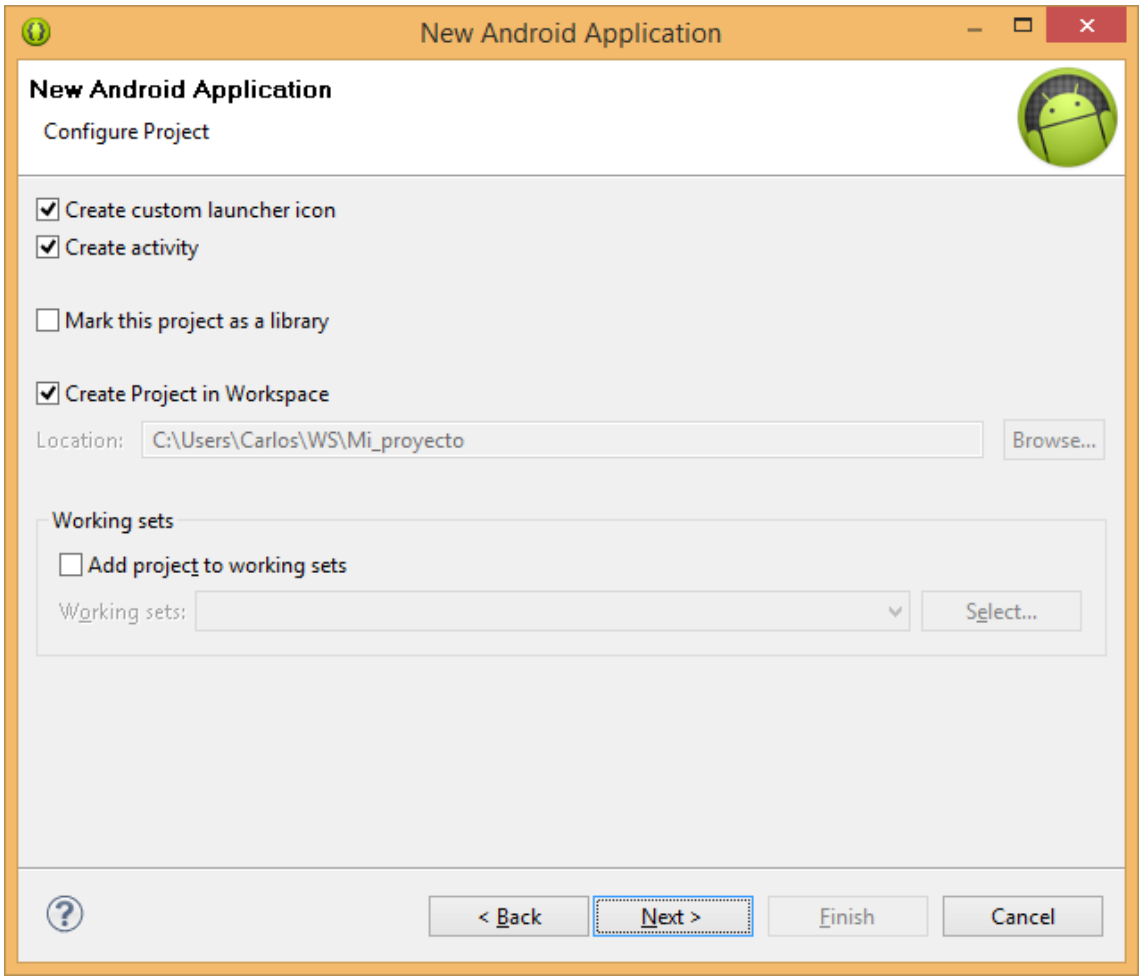


Figura 5. Configuración.

Posteriormente podemos personalizar la aplicación con un icono propio o seleccionar uno de los que ya existe en Eclipse (figura 6). Pinchamos el botón Next.

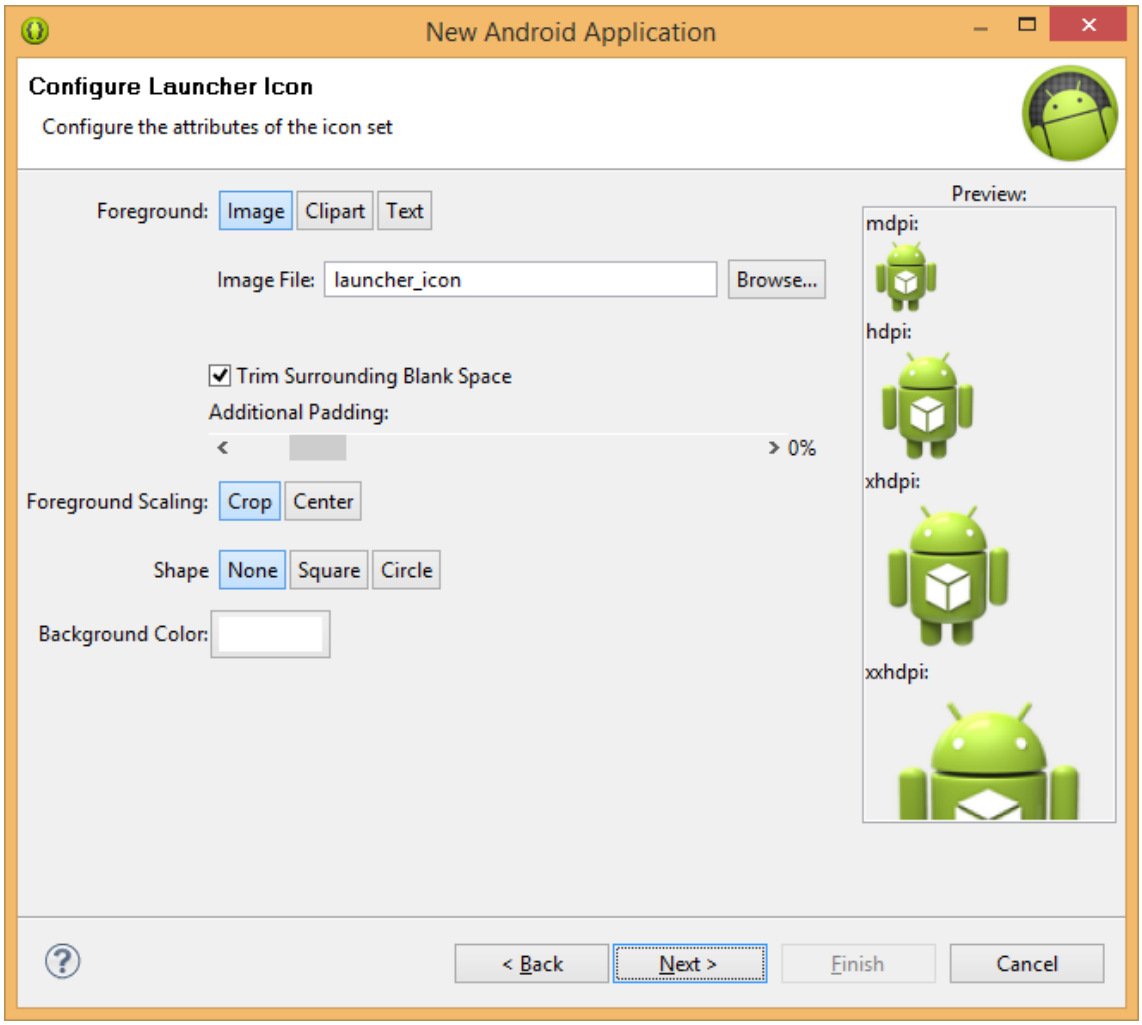


Figura 6. Icono de la aplicación.

Después tenemos la posibilidad de crear la primera actividad de la aplicación, y se nos presentan distintas opciones para la misma (figura 7). Pinchamos el botón Next.

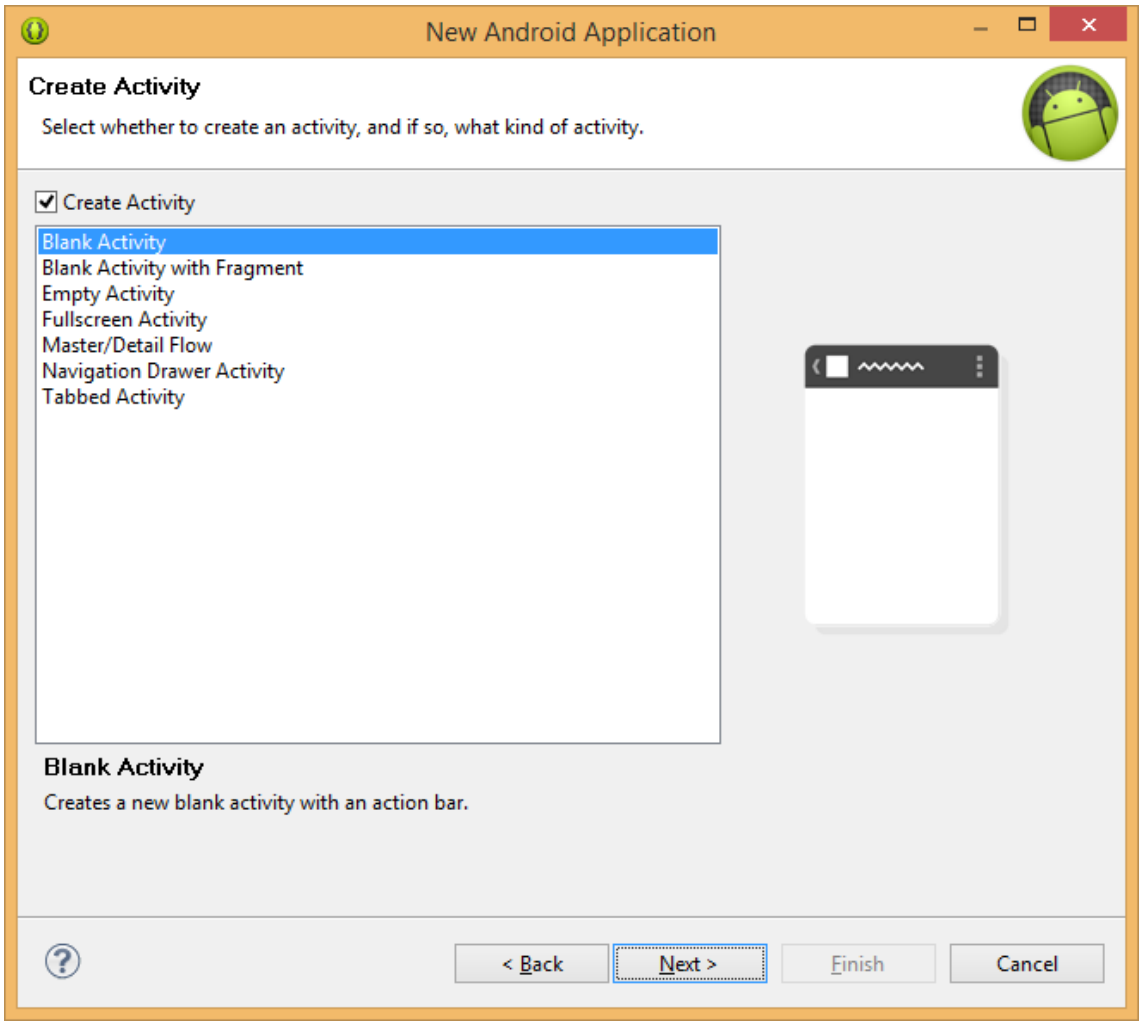


Figura 7. Crear una actividad.

Rellenamos ahora los nombres de la actividad principal y de su layout, lo normal es dejarlo tal y como está (figura 8). Pinchamos el botón Finish.

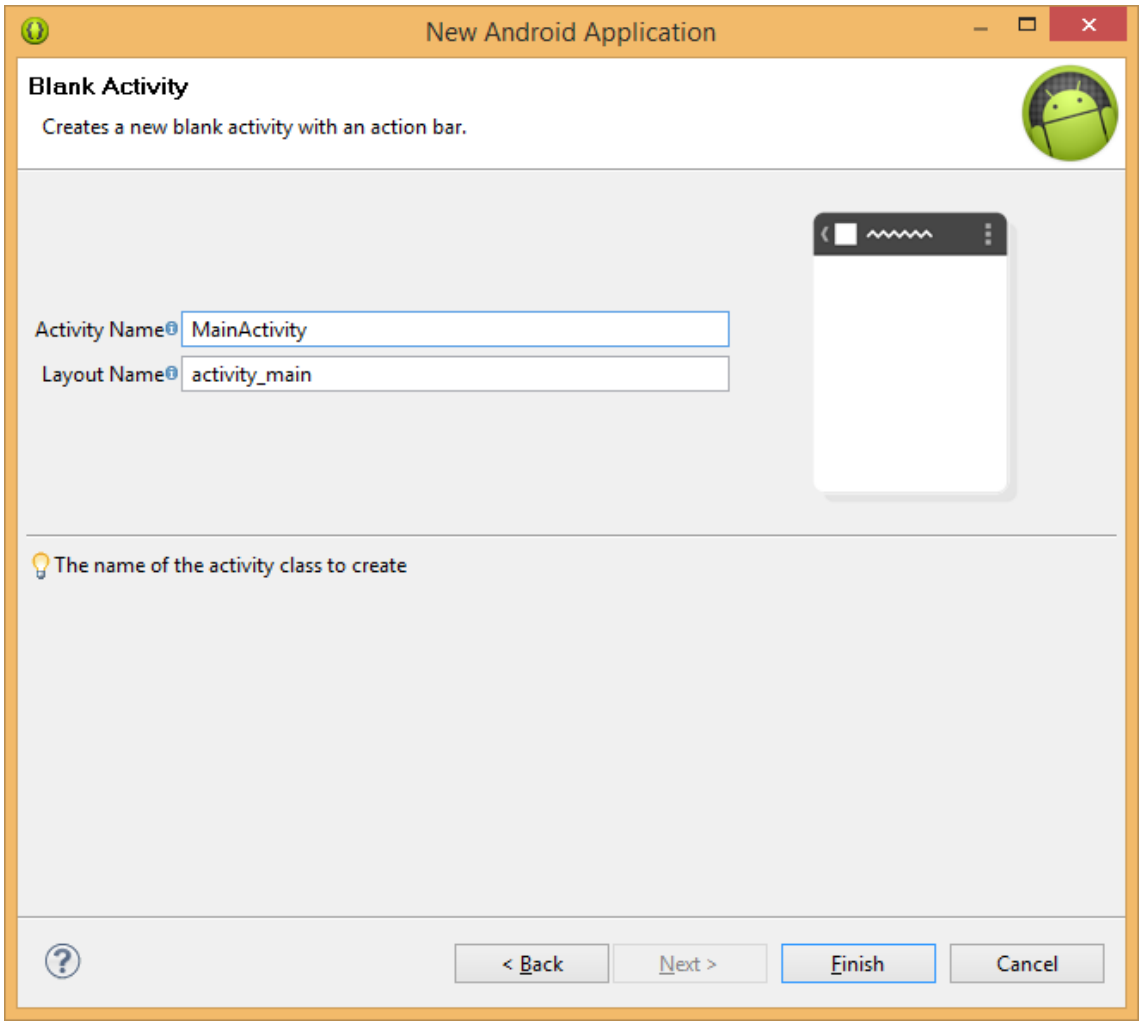


Figura 8. Datos de la actividad principal.

Tenemos ya el nuevo proyecto creado que cuenta con una actividad (la principal). Por defecto la actividad principal se genera con el código necesario para mostrar en la pantalla el clásico mensaje de "¡Hola mundo!", en este caso en inglés "Hello world!" (figura 9). Desde este punto ya podemos comenzar a editar la aplicación, tanto el código Java (fichero MainActivity.java), como el/los ficheros de configuración de la interfaz gráfica de dicha actividad (activity_main.xml).

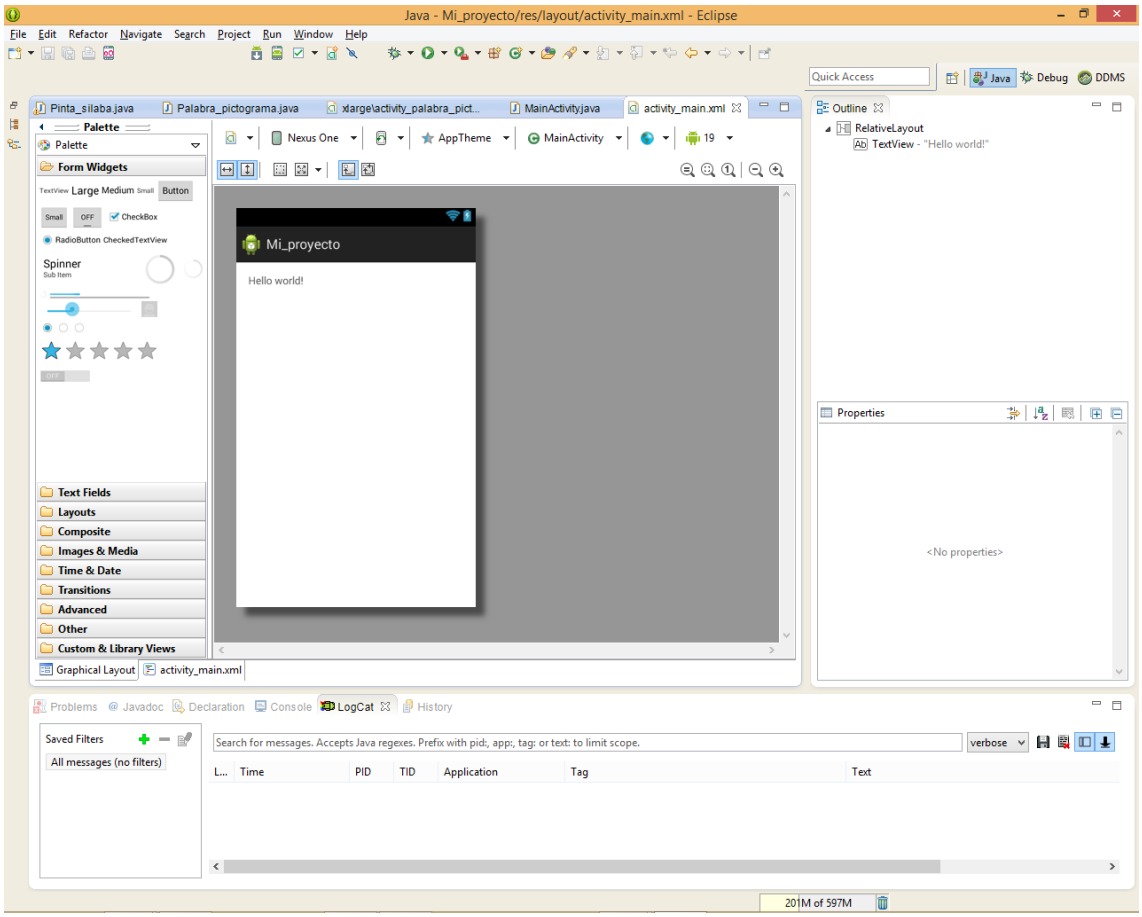


Figura 9. Proyecto creado, actividad principal.

4.13 Pruebas y depuración de la aplicación

En el proceso de desarrollo de una aplicación siempre es conveniente ir realizando periódicamente pruebas de funcionamiento para comprobar la idoneidad del código utilizado. Estás pruebas se realizarán de forma discrecional, según se vayan superando distintos hitos que el programador deberá tener claros. Por ejemplo, si la aplicación consta de diferentes actividades, se deberían probar éstas en el momento de finalización de cada una de ellas, e intentar solucionar los fallos uno a uno según se vayan sucediendo. Sería una mala idea dejar las pruebas para el final, cuando ya se hayan codificado todas las actividades pues podría tornarse en un trabajo muy largo y tedioso, que podría haber sido más llevadero y efectivo, probando y depurando como ya hemos dicho, por hitos superados. Además de ésta última forma, tenemos mucho más fresco todo lo relacionado con cada aplicación que si esperamos a implementar las pruebas al final.

Para probar la aplicación, habrá que ejecutarla, bien sea en un dispositivo físico, o en un emulador (como mencionamos anteriormente). En cualquier caso el procedimiento será el mismo. Nos situaremos sobre el explorador de paquetes (zona izquierda de Eclipse), colocaremos el cursor del ratón sobre el proyecto que deseamos probar (Mi_proyecto), y presionaremos el botón derecho del ratón para abrir el menú contextual. Nos movemos de la siguiente manera por dicho menú Run As > Android Application (figura 10). Android comenzará el proceso de compilado para comprobar que no existan errores en el código y en

caso de éxito, instalará la aplicación o bien en el dispositivo físico conectado al ordenador, o bien en el emulador que estemos utilizando (figura 11).

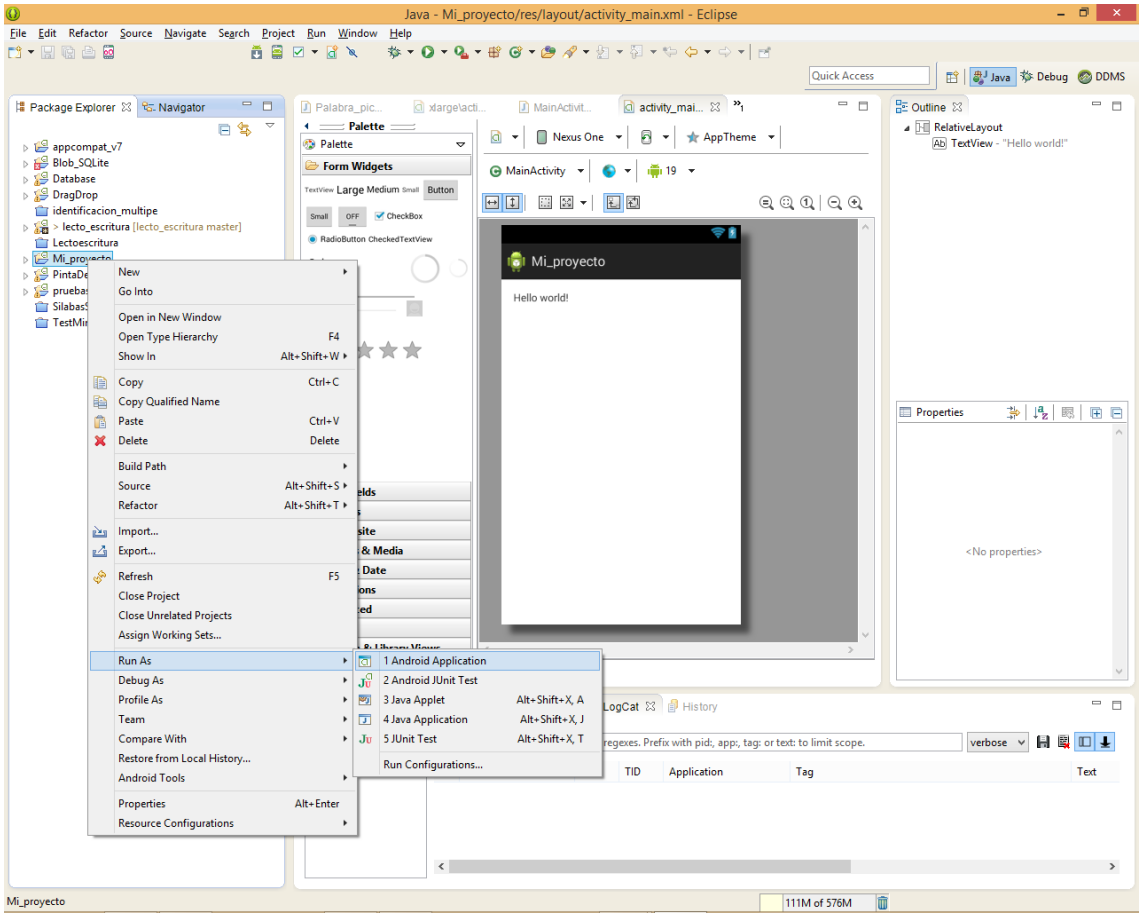


Figura 10. Ejecución de la aplicación.

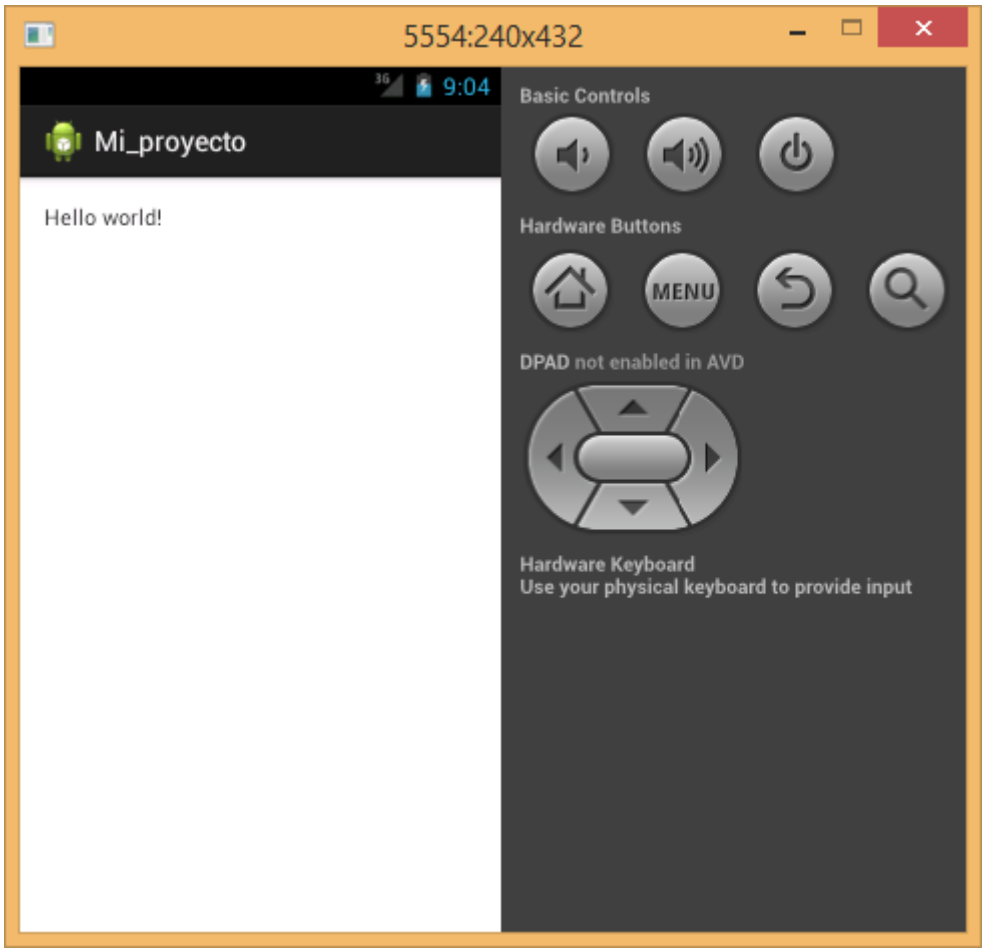


Figura 11. Aplicación corriendo en emulador.

Con la aplicación en ejecución, debemos realizar todas las pruebas posibles para detectar fallos de programación. En caso de encontrar cualquier error, existe una función muy interesante en Eclipse para la depuración (debugging) de código. Para acceder a ella, hay que seleccionar la pestaña Debug situada en la parte superior derecha de Eclipse, junto a la pestaña de Java que es la que se encuentra activa por defecto (figura 12).

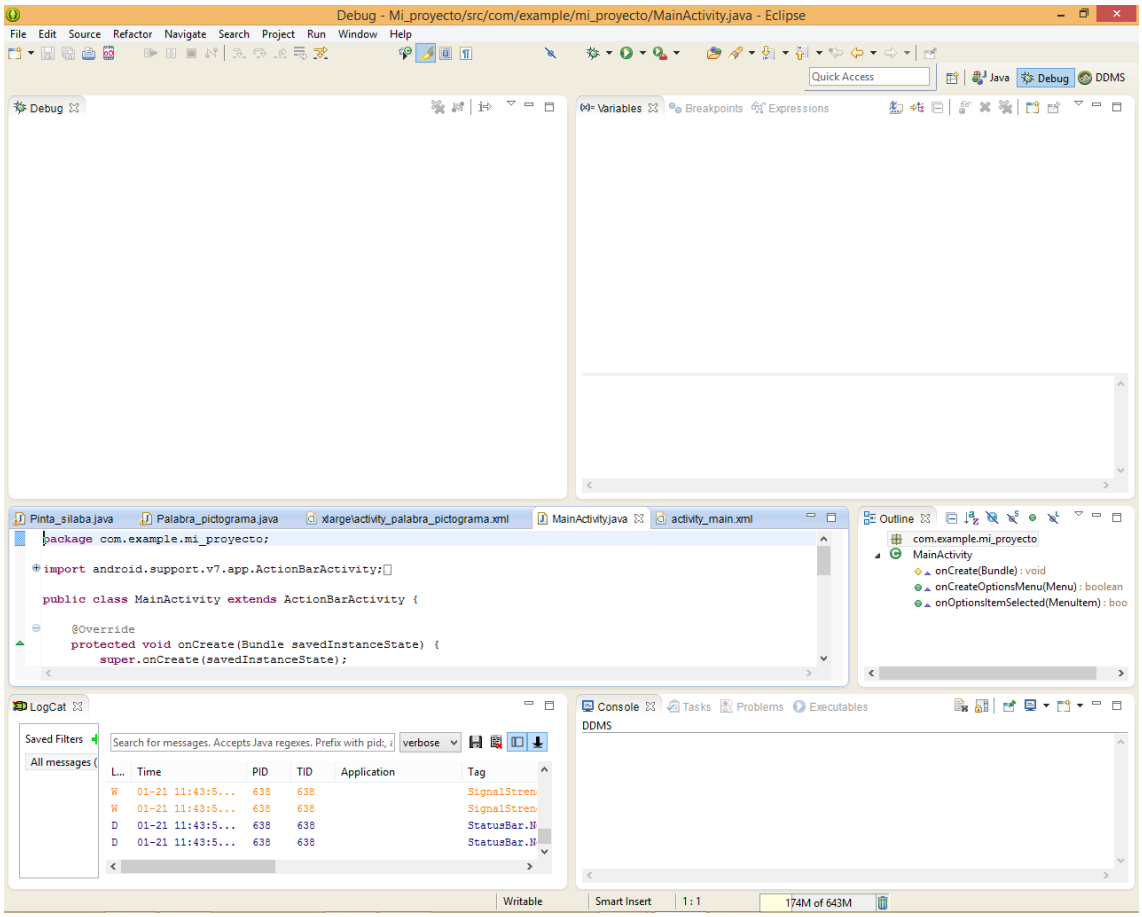


Figura 12. Paneles de depuración.

Desde los paneles de depuración se podrán establecer tantos puntos de ruptura en las líneas de código como sean necesarios. Estos puntos de ruptura provocan una pausa en la ejecución de la aplicación justo en la línea donde se han situado cuando se ejecuta dicha aplicación en modo depuración. De esta manera y a partir de ese punto, se puede comprobar una por una las líneas de código de la parte de la aplicación que está produciendo el fallo y analizar así cual puede ser la causa del error. Además de ejecutar una por una las líneas de código (pulsando la tecla F6 se salta a la siguiente línea de código), tenemos la posibilidad de comprobar los valores asignados a variables y otros recursos, lo que nos pueden ayudar a seguir la pista de problema (punteros nulos, valores inadecuados, etc.). En cualquier momento se puede reanudar la ejecución normal de la aplicación (pulsando F8), que solo será interrumpida si se vuelve a encontrar con otro punto de ruptura, si lo hubiera, donde se repetirá el procedimiento anterior. En cualquier momento se puede abortar la función depuración, pasando a ejecución normal (pulsando el icono Disconnect).

Para ejecutar una aplicación en modo depuración hay que situar el cursor sobre el proyecto de la aplicación a depurar y pulsar el botón derecho del ratón. Aparecerá el menú contextual de la figura 13, donde se debe seguir la secuencia Debug As > Android Application, y seleccionar el dispositivo de ejecución.

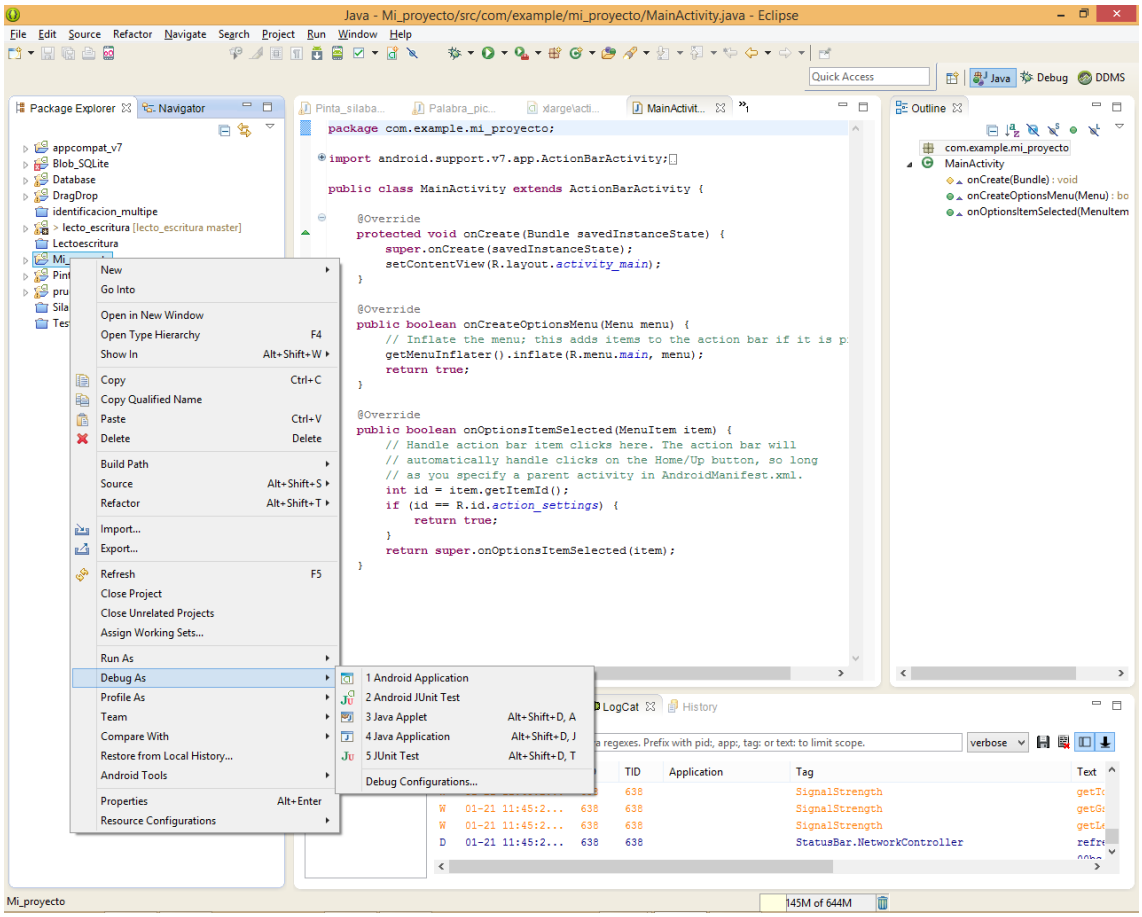


Figura 13. Ejecución de la depuración.

4.14 SQLITE

SQLite es un sistema de gestión de bases de datos (SGBD) relacional de código libre compatible con ACID (Atomicity, Consistency, Isolation and Durability: Atomicidad, Consistencia, Aislamiento y Durabilidad), contenida en una relativamente pequeña (~275 kiB) biblioteca escrita en C. SQLite es un proyecto de dominio público creado por D. Richard Hipp.

A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción.

En su versión 3, SQLite permite bases de datos de hasta 2 Terabytes de tamaño, y también permite la inclusión de campos tipo BLOB (utilizados en esta aplicación para almacenar imágenes, aunque podrían guardar también vídeos y audios).

4.14.1 Características

La biblioteca implementa la mayor parte del estándar SQL-92, incluyendo transacciones de base de datos atómicas, consistencia de base de datos, aislamiento, y durabilidad (ACID), triggers y la mayor parte de las consultas complejas.

SQLite usa un sistema de tipos inusual. En lugar de asignar un tipo a una columna como en la mayor parte de los sistemas de bases de datos SQL, los tipos se asignan a los valores individuales. Por ejemplo, se puede insertar un string en una columna de tipo entero (a pesar de que SQLite tratará en primera instancia de convertir la cadena en un entero). Algunos usuarios consideran esto como una innovación que hace que la base de datos sea mucho más útil, sobre todo al ser utilizada desde un lenguaje de scripting de tipos dinámicos. Otros usuarios lo ven como un gran inconveniente, ya que la técnica no es portable a otras bases de datos SQL. SQLite no trataba de transformar los datos al tipo de la columna hasta la versión 3.

Varios procesos o hilos pueden acceder a la misma base de datos sin problemas. Varios accesos de lectura pueden ser servidos en paralelo. Un acceso de escritura sólo puede ser servido si no se está sirviendo ningún otro acceso concurrentemente. En caso contrario, el acceso de escritura falla devolviendo un código de error (o puede automáticamente reintentarse hasta que expira un tiempo de expiración configurable). Esta situación de acceso concurrente podría cambiar cuando se está trabajando con tablas temporales. Sin embargo, podría producirse un interbloqueo debido al multihilo. Este punto fue tratado en la versión 3.3.4, desarrollada el 11 de febrero de 2006.

Existe un programa independiente de nombre SQLite que puede ser utilizado para consultar y gestionar los ficheros de base de datos SQLite. También sirve como ejemplo para la escritura de aplicaciones utilizando la biblioteca SQLite.

Algunas características distintivas de SQLite son:

- No necesita configuración, ni tras la instalación inicial ni para el posterior mantenimiento.
- No utiliza servidor. Se puede utilizar embebida dentro de otra aplicación o gestionar los ficheros de datos a través de una pequeña aplicación de consola descargable desde su web.
- Utiliza un sólo fichero para almacenar los datos. Una base de datos de SQLite se compone de un único fichero que puede almacenarse en cualquier ruta de la máquina.
- Los ficheros de datos son multiplataforma. Una base de datos creada en una máquina y sistema operativo concreto puede copiarse y utilizarse bajo cualquier otra plataforma.
- Es muy compacta. La librería que se integra con otras aplicaciones ocupa unos 200 KBytes.
- Utiliza tipado dinámico (Manifest Typing). SQLite permite almacenar cualquier valor de cualquier tipo en cualquier registro de una tabla de la base de datos, independientemente del tipo declarado al crear la tabla.
- Utiliza registros de longitud variable. Cada dato almacenado en la base de datos ocupará su tamaño real y no el reservado según su tipo.

4.14.2 Lenguajes de programación

La biblioteca puede ser usada desde programas en C/C++, aunque están disponibles enlaces para Tcl y muchos otros lenguajes de programación interpretado.

SQLite se encuentra embebido en el REALbasic framework, haciendo posible que aplicaciones desarrolladas en REALbasic para Windows, Linux o Mac OS X usen la base de datos SQLite.

Existe un módulo DBI/DBD para Perl disponible en CPAN, DBD::SQLite, no es una interface para SQLite, sino que incluye el motor completo de SQLite en sí mismo por lo cual no necesita ningún software adicional.

Python incluye soporte para SQLite nativamente desde la versión 2.5 incorporado en la Biblioteca Estándar como el módulo sqlite3.3 Para versiones anteriores de Python, el módulo no está incorporado y debe instalarse (su nombre es PySQLite).

Hay otro módulo para Visual Basic 6 llamado VBSqlite

Desde Delphi se puede usar SQLite a través de los componentes libres ZeosLib.

PHP incluye SQLite, desde la versión 5. SQLite también funciona con PHP 4 pero no viene incluido en él. Para más detalles vea el manual y PECL info.

Desde Java se puede acceder mediante el driver de SQLite JDBC

Desde .NET se puede acceder usando el proyecto de código abierto System.Data.SQLite

Desde Lazarus 0.9.8 y Free Pascal 2.0.0, SQLite está disponible para programadores de Pascal. Tutorial: «Lazarus Database Tutorial, Lazarus and SQLite» (en inglés).

Mac OS X v10.4 incluye SQLite, y es una de las opciones en la Core Data API de Apple. AppleScript puede abrir, crear, y manipular base de datos SQLite por medio de la aplicación de ayuda "Database Events" de Mac OS X 10.4.

BlitzMAX posee un MOD que permite trabajar con bases de datos SQLite.

El componente de base de datos (gb.db) de Gambas soporta SQLite en sus versiones 1, 2 y 3

El lenguaje de programación de vídeo juegos Bennu tiene un mod de SQlite disponible

El lenguaje de programación de scripting para Windows AutoIt v.3.x a través de la DLL SQLite.dll.

4.14.3 Software que utiliza SQLite

SQLite es utilizado en una gran variedad de aplicaciones, destacando las siguientes:

- Adobe Photoshop Elements utiliza SQLite como motor de base de datos en su última versión del producto (la 6.0) en sustitución del Microsoft Access, utilizado en las versiones anteriores.
- Clementine usa SQLite para guardar su colección de datos por defecto.
- Kexi usa SQLite como un motor de base de datos interno por defecto.

- Mozilla Firefox usa SQLite para almacenar, entre otros, las cookies, los favoritos, el historial y las direcciones de red válidas.
- Los desarrolladores de OpenOffice.org han considerado incluir SQLite en el modelo de base de datos de Base, pero esto depende en gran manera del progreso de sqlite-sdbc-driver, que está todavía en estado de alpha. Actualmente han decidido usar HSQLDB.
- Varias aplicaciones de Apple utilizan SQLite, incluyendo Apple Mail y el gestor de RSS que se distribuye con Mac OS X. El software Aperture de Apple guarda la información de las imágenes en una base de datos SQLite, utilizando la API Core Data.
- El navegador web Opera usa SQLite para la gestión de bases de datos WebSQL.
- Skype es otra aplicación de gran despliegue que utiliza SQLite.
- SQLFilter, un plugin para OmniPeek, usa SQLite para indexar paquetes en una base de datos para poder ser consultada por medio de SQL.
- The New Yorker guarda el índice para un set de DVD conteniendo todos los números publicados por la revista.
- XBMC Media Center (antes conocido como "XBox Media Center") es un reproductor de medios de audio, video, fotos, etc., de código libre (open source) multi-plataforma a la vez que un centro de entretenimiento. Usa SQLite para administrar las librerías de música, video y fotografías, listas de reproducción y bookmarks entre otras utilidades menores.

Debido a su pequeño tamaño, SQLite es muy adecuado para los sistemas integrados, pero también está incluido en:

- Android
- BlackBerry
- Windows Phone 8
- Google Chrome
- iOS
- Firefox OS
- Maemo
- MeeGo
- Symbian OS
- webOS

4.14.4 SQLite y Android

Android tiene SQLite preinstalado, por lo que no es necesario llevar a cabo ningún tipo de instalación. Android incorpora una serie de herramientas necesarias para la creación y gestión de bases de datos SQLite, así como una completa API para llevar a cabo de manera sencilla todas las tareas necesarias.

En Android existe un conjunto de clases Java que interactúan directamente con el sistema gestor de base de datos de SQLite:

- **SQLiteOpenHelper**. Utilizada para la creación y modificación de bases de datos. En realidad la creación, actualización y conexión a una base de datos SQLite se realizará a través de una clase propia derivada de SQLiteOpenHelper. Esta clase debe contener los métodos onCreate() y onUpgrade() donde definiremos la forma de crear una base de datos y la actualización de su estructura. El método onCreate() será ejecutado

automáticamente cuando sea necesaria la creación de la base de datos, es decir, cuando aún no exista. El método onUpgrade() se ejecutará automáticamente cuando sea necesaria una actualización de la estructura de la base de datos o una conversión de los datos.

- **Cursor**. La clase Cursor otorga acceso a los resultados de una consulta de base de datos y permite recorrer de uno en uno estos resultados. Estos resultados serán accesibles desde el código de la aplicación.
  Un cursor es una colección de filas. Se usa el método moveToFirst() antes de leer cualquier dato del cursor ya que éste comienza posicionado antes de la primera fila. Es necesario conocer los nombres de las columnas así como sus tipos. Los cursores son del tipo aleatorio, es decir, podemos movernos por él hacia delante, hacia atrás o saltar de una posición a otra. Los cursores tienen otros métodos que nos permiten navegar por ellos.

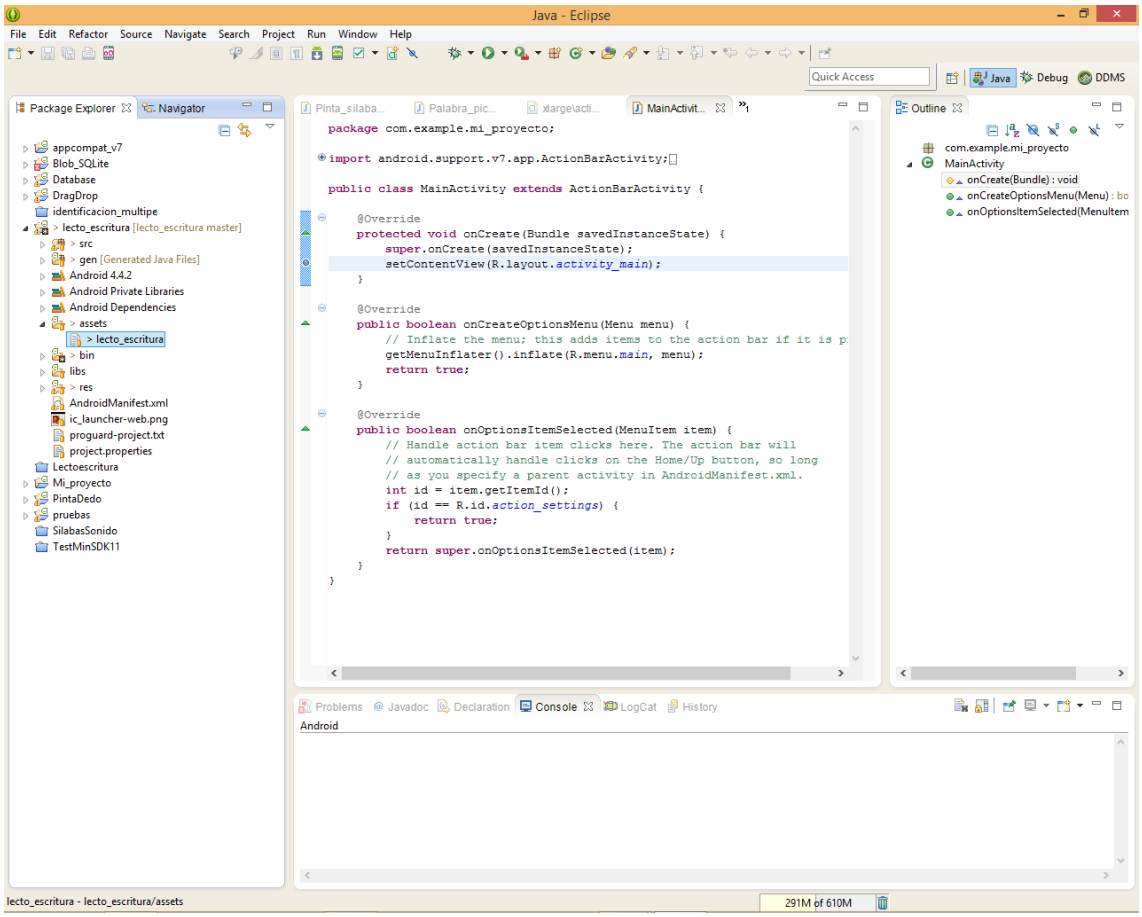
- **SQLiteDatabase.** Esta clase es la verdadera interfaz entre el código de una aplicación y la base de datos SQLite. Incluye funciones para realizar las operaciones basadas en SQL como INSERT, DELETE, QUERY y RAWQUERY (una sentencia de consulta SQL que devuelve los resultados en forma de objeto Cursor).

Las bases de datos SQLite creadas por aplicaciones Android se almacenan en la memoria del teléfono en un fichero con el mismo nombre de la base de datos situado en la siguiente ruta:

/data/data/paquete.de.aplicacion/databases/nombre_basedatos

Las bases de datos SQLite se pueden crear en la aplicación de dos maneras. Bien de en forma de instrucciones (código) de la propia aplicación, con las posibilidades que ofrecen algunas de las clases vistas en este apartado, o bien existe la posibilidad de pasar a la aplicación una base de datos ya existente, creada previamente a la ejecución de la aplicación (por ejemplo en un PC). De esta manera se facilita enormemente la creación de una base de datos y la incorporación de contenidos. Éste último método ha sido el utilizado en la aplicación que nos ocupa. Se han insertado los valores de los registros por medio de un PC posteriormente se ha pasado el fichero de la base de datos SQLite a la carpeta assets del proyecto de la aplicación, como se puede ver en la figura 14 (lecto_escritura). Una vez incorporado la base de datos al directorio correspondiente, en el momento de la instalación y ejecución de la aplicación en el dispositivo móvil o emulador, esta será transferida al directorio correspondiente que sigue la ruta apuntada un poco más arriba:

/data/data/paquete.de.aplicacion/databases/lecto_escritura



14. Base de datos SQLite

4.15 Eclipse IDE

Eclipse es una plataforma informática compuesta por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse.

Eclipse es también una comunidad de usuarios, extendiendo constantemente las áreas de aplicación cubiertas. Un ejemplo es el recientemente creado Eclipse Modeling Project, cubriendo casi todas las áreas de Model Driven Engineering.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

Eclipse fue liberado originalmente bajo la Common Public License, pero después fue re-licenciado bajo la Eclipse Public License.

La base para Eclipse es la Plataforma de cliente enriquecido (del Inglés Rich Client Platform RCP). Los siguientes componentes constituyen la plataforma de cliente enriquecido:

- Plataforma principal - inicio de Eclipse, ejecución de plugins
- OSGi - una plataforma para bundling estándar.
- El Standard Widget Toolkit (SWT) - Un widget toolkit portable.
- JFace - manejo de archivos, manejo de texto, editores de texto
- El Workbench de Eclipse - vistas, editores, perspectivas, asistentes

Los widgets de Eclipse están implementados por una herramienta de widget para Java llamada Standard Widget Toolkit, a diferencia de la mayoría de las aplicaciones Java, que usan las opciones estándar Abstract Window Toolkit (AWT) o Swing. La interfaz de usuario de Eclipse también tiene una capa GUI intermedia llamada JFace, la cual simplifica la construcción de aplicaciones basadas en SWT.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de cliente enriquecido, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos. La arquitectura plugin permite escribir cualquier extensión deseada en el entorno, como sería Gestión de la configuración. Se provee soporte para Java y CVS en el SDK de Eclipse. Y no tiene por qué ser usado únicamente con estos lenguajes, ya que soporta otros lenguajes de programación.

La definición que da el proyecto Eclipse acerca de su software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular".

En cuanto a las aplicaciones clientes, Eclipse provee al programador con frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc. Por ejemplo, GEF (Graphic Editing Framework - Framework para la edición gráfica) es un plugin de Eclipse para el desarrollo de editores visuales que pueden ir desde procesadores de texto WYSIWYG (What You See Is What You Get) hasta editores de diagramas UML (Unified Modeling Languaje), interfaces gráficas para el usuario (GUI), etc. Dado que los editores realizados con GEF "viven" dentro de Eclipse, además de poder ser usados conjuntamente con otros plugins, hacen uso de su interfaz gráfica personalizable y profesional.

El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. Mediante diversos plugins estas herramientas están también disponibles para otros lenguajes como C/C++ (Eclipse CDT) y en la medida de lo posible para lenguajes de script no tipados como PHP o Javascript. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadatos en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.

Eclipse dispone de un Editor de texto con resaltado de sintaxis. La compilación es en tiempo real. Tiene pruebas unitarias con JUnit, control de versiones con CVS, integración con Ant, asistentes (wizards) para creación de proyectos, clases, tests, etc., y refactorización.

Asimismo, a través de "plugins" libremente disponibles es posible añadir control de versiones con Subversion e integración con Hibernate.

5. Discapacidad

Según la OMS, discapacidad es un término general que abarca las deficiencias, las limitaciones de la actividad y las restricciones de la participación. Las deficiencias son problemas que afectan a una estructura o función corporal; las limitaciones de la actividad son dificultades para ejecutar acciones o tareas, y las restricciones de la participación son problemas para participar en situaciones vitales.

Por consiguiente, la discapacidad es un fenómeno complejo que refleja una interacción entre las características del organismo humano y las características de la sociedad en la que vive[4].

En el sistema educativo español, los niños con discapacidad se engloban dentro del término ACNEE (Alumnos Con Necesidades Educativas Especiales). Según la LOE (Ley Orgánica de la Enseñanza, Ley Orgánica 2/2006, de 3 de mayo, de Educación, modificada por la LOMCE, Ley Orgánica 8/2013, de 9 de diciembre, para la Mejora de la Calidad Educativa.), se entiende por alumnado que presenta necesidades educativas especiales, aquel que requiera, por un periodo de su escolarización o a lo largo de toda ella, determinados apoyos y atenciones educativas específicas derivadas de discapacidad o trastornos graves de conducta[5].

Se puede hacer la siguiente clasificación dentro de los ACNEE:

- **Discapacidad física**
- **Discapacidad psíquica**
- **Discapacidad auditiva, visual**
- **Necesidades del lenguaje**
- **Trastornos del espectro autista**
- **Graves alteraciones de la conducta**
- **Altas capacidades**

A su vez se pueden abrir nuevas clasificaciones dentro de cada una de las anteriores

- **Discapacidad física o psíquica:** leve, moderada, grave, profunda
- **Discapacidad motora:** parálisis cerebral, espina bífida, trastornos del crecimiento, otras lesiones.
- **Discapacidad auditiva:** hipoacusia, sordera profunda
- **Discapacidad visual:** baja visión, ceguera
- **Trastornos graves del lenguaje:** disfasia, afasia
- **Trastornos del espectro autista:** autismo, síndrome de Asperger, síndrome de Rett, trastorno desintegrativo infantil, trastorno del desarrollo no especificado.

5.1 Definición de NEE

Entendemos por Necesidades Educativas Especiales (NEE), al conjunto de medidas pedagógicas que se ponen en marcha para compensar las dificultades que presenta un alumno al acceder al currículo que le corresponde por edad.

Dichas dificultadas son superiores al resto de los alumnos, por diversas causas: discapacidades, trastornos graves de conducta, altas capacidades intelectuales o por integración tardía en el sistema educativo.

5.2 Tipos de discapacidad

Toda discapacidad tiene su origen en una o varias deficiencias funcionales o estructurales de algún órgano corporal, y en este sentido se considera como deficiencia cualquier anomalía de un órgano o de una función propia de ese órgano con resultado discapacitante.

Partiendo de esta distinción básica promovida por la OMS a través de la Clasificación Internacional del Funcionamiento, de la Discapacidad y de la Salud (CIF), se puede identificar numerosas clases de deficiencia asociadas a las distintas discapacidades.

Para identificar las características de los grandes grupos se ha recurrido a esa misma fuente agrupando las deficiencias en las categorías física, mental y sensorial, teniendo siempre presente la gran heterogeneidad que existe dentro de cada uno.

- **Las deficiencias físicas:** se considerará que una persona tiene deficiencia física cuando padezca anomalías orgánicas en el aparato locomotor o las extremidades (cabeza, columna vertebral, extremidades superiores y extremidades inferiores). También se incluirán las deficiencias del sistema nervioso, referidas a las parálisis de extremidades superiores e inferiores, paraplejías y tetraplejías y a los trastornos de coordinación de los movimientos, entre otras. Un último subconjunto recogido en la categoría de discapacidades físicas es el referido a las alteraciones viscerales, esto es, a los aparatos respiratorio, cardiovascular, digestivo, genitourinario, sistema endocrino-metabólico y sistema inmunitario.
  En todos los casos de deficiencias de carácter físico el eje problemático en torno al cual se estructura la trama de la integración (deficitaria) es la autonomía personal, ya que aunque en cada etapa del ciclo vital las expectativas en torno a la autonomía son distintas, como también lo son entre las personas que no padecen discapacidad, se trata de un elemento esencial desde el punto de vista de la calidad de vida. Pues bien, hablar de autonomía supone referirse a ámbitos tan variados como el laboral, el educativo, la comunicación social y por supuesto la accesibilidad, que aglutina a todas estas facetas vitales. La escasa participación en actividad y empleo, el déficit y el desajuste educativo, así como la sobreprotección familiar, que redundan en la falta de autonomía, son problemas comunes a todas las personas con discapacidad. Pero tal vez se manifiesten de forma especialmente reconocible en términos de accesibilidad en aquellas personas que tienen muy reducida su capacidad de movimiento, como los usuarios de sillas de ruedas.

- **Las deficiencias mentales: c**omo ocurre con el resto de los colectivos, el integrado por las personas con deficiencias mentales es de difícil cuantificación, entre otras razones por la falta de precisión en la determinación de sus límites. Concretamente la EDDES (Encuesta sobre Discapacidades, Deficiencias y Estado de Salud) incluye en la categoría de deficiencia mental el espectro del retraso mental en sus grados severo, moderado y leve, además del retraso madurativo, las demencias y otros trastornos mentales. En esta última recoge trastornos tan diversos como el autismo, las esquizofrenias, los trastornos psicóticos, somáticos y de la personalidad, entre otros. La falta de acuerdo en torno a la idoneidad de la inclusión de algunos de éstos últimos en categorías distintas y sobre todo la imposibilidad de realizar su cuantificación de manera aislada, hace ineludible una exploración previa del conjunto, considerando la categoría 'otros trastornos' como enfermedades mentales.
- **Las deficiencias sensoriales:** al igual que en los casos de los grupos anteriores, al hablar de deficiencias sensoriales es necesario recordar que las categorías de análisis presentan limitaciones en la descripción de la realidad para la que se aplican. No obstante, asumiendo los criterios de clasificación empleados en la encuesta, son útiles como aproximación al tamaño y sobre todo a la composición del colectivo resultante. La categoría 'deficiencias sensoriales' incluye, para los fines de este estudio a quienes presentan trastornos relacionados con la vista y el oído. Dentro del grupo de las deficiencias sensoriales se incluyen, como se ha dicho, colectivos afectados por trastornos de distinta naturaleza. Las deficiencias auditivas presentan a su vez distintos grados, desde las hipoacusias (mala audición) de carácter leve hasta la sordera total prelocutiva y postlocutiva, y los trastornos relacionados con el equilibrio. A estas diferencias se unen las distintas estrategias técnicas y comunicativas empleadas por quienes padecen deficiencias auditivas (lenguaje de signos, implantes cocleares o audífonos), configurando un colectivo de rasgos muy heterogéneos, tanto por sus perfiles orgánicos como por sus estrategias de integración. El otro gran colectivo incluido en la categoría de deficiencias sensoriales lo constituyen las personas con trastornos visuales.
- **Trastornos del espectro autista:** trastornos caracterizados por alteraciones cualitativas en la interacción social, la comunicación y la imaginación, así como por la presencia de patrones estereotipados de conductas e intereses. A efectos de clasificación, se considerarán cinco grupos:
- **Trastornos graves de conducta:** patrones de comportamientos inadaptados y persistentes en al menos dos ámbitos distintos de socialización, que implican un deterioro del funcionamiento diario e incontrolabilidad manifiesta de los comportamientos por parte de las personas encargadas de su cuidado y educación. Repercuten en el propio desarrollo y generan consecuencias negativas para sí mismo y/o para los demás, y requieren intervenciones multidisciplinares y coordinación intersectorial. A efectos de clasificación, se considerarán tres grupos: trastorno disocial, trastorno por negativismo desafiante y trastorno de comportamiento perturbador no especificado.

5.3 Definiciones de discapacidades

- **Parálisis cerebral:** La parálisis cerebral es un trastorno permanente y no progresivo que afecta a la psicomotricidad del paciente. En un nuevo consenso internacional, se

propone como definición: "La parálisis cerebral describe un grupo de trastornos del desarrollo psicomotor, que causan una limitación de la actividad de la persona, atribuida a problemas en el desarrollo cerebral del feto o del niño. Los desórdenes psicomotrices de la parálisis cerebral están a menudo acompañados de problemas sensitivos, cognitivos, de comunicación y percepción, y en algunas ocasiones, de trastornos del comportamiento". Las lesiones cerebrales de la PC ocurren desde el período fetal hasta la edad de 3 años. Los daños cerebrales después de la edad de 3 años hasta el período adulto pueden manifestarse como PC, pero, por definición, estas lesiones no son PC. Hay autores que recomiendan, en determinados casos, no establecer el diagnóstico de PC hasta los 5 años de edad.

La incidencia de esta condición en países desarrollados es de aproximadamente 2 – 2,5 por cada mil nacimientos. Esta incidencia no ha bajado en los últimos 60 años a pesar de los avances médicos como la monitorización de las constantes vitales de los fetos. La Parálisis cerebral no tiene cura conocida; la intervención médica aparece como una ayuda. Estos tratamientos para el desarrollo personal del paciente se introducen en su vida diaria hasta su muerte.

La parálisis cerebral es un término que agrupa un grupo de diferentes condiciones. Hay que tener en cuenta que no hay dos personas con parálisis cerebral con las mismas características o el mismo diagnóstico.

- **Espina bífida:** La espina bífida es una malformación congénita en la que existe un cierre incompleto del tubo neural al final del primer mes de vida embrionaria y posteriormente, el cierre incompleto de las últimas vértebras.

  La principal causa de la espina bífida es la deficiencia de ácido fólico en la madre durante los meses previos al embarazo y en los tres meses siguientes, aunque existe un 5% de los casos cuya causa es desconocida. Ya hoy en día se ha comprobado que la espina bífida no tiene un componente hereditario. Lo que se heredaría sería la dificultad de la madre para procesar el ácido fólico, lo que ocurre en muy pocos casos. También se comprobó que una persona con espina bífida no tendrá necesariamente hijos con la misma discapacidad.

  Básicamente existen dos tipos de espina bífida: la espina bífida oculta y la espina bífida abierta o quística.
- **Hipoacusia:** La hipoacusia es la pérdida parcial de la capacidad auditiva. Esta pérdida puede ser desde leve o superficial hasta moderada, y se puede dar de manera unilateral o bilateral dependiendo de que sea en uno o ambos oídos; esta pérdida puede ser de más de 40 decibelios en adelante. Las personas con hipoacusia habitualmente utilizan el canal auditivo y el lenguaje oral para comunicarse. Se benefician del uso de auxiliares auditivos para recuperar hasta en un 20 a 30 % de la audición.

- **Sordera:** La sordera es la dificultad o la imposibilidad de usar el sentido del oído debido a una pérdida de la capacidad auditiva parcial (hipoacusia) o total (cofosis), y unilateral o bilateral. Así pues, una persona sorda será incapaz o tendrá problemas para escuchar. Ésta puede ser un rasgo hereditario o puede ser consecuencia de una enfermedad, traumatismo, exposición a largo plazo al ruido, o medicamentos agresivos para el nervio auditivo.
  Hoy en día hay algunos términos mal utilizados para referirnos a las personas sordas o a la lengua de signos:
  - Sordomudo: es una persona que tiene un problema de audición y de cuerdas vocales. Una persona sorda no es muda, ya que tiene la capacidad de hablar.
  - Lenguaje de signos: Es Lengua de Signos ya que el lenguaje es la capacidad, la facultad comunicativa propia del ser humano y la lengua es el sistema de comunicación de los humanos.
- **Ceguera:** La ceguera es una discapacidad física que consiste en la pérdida total o parcial del sentido de la vista. Existen varios tipos de ceguera parcial dependiendo del grado y tipo de pérdida de visión, como la visión reducida, el escotoma, la ceguera parcial (de un ojo) o el daltonismo.

  - Ceguera Parcial: es cuando la persona ve con baja visión o no tiene la suficiente capacidad de tener una buena visión y se ven obligados a usar gafas para tener buena visión.
  - Ceguera Total o Completa: es cuando la persona no ve ni siente absolutamente nada, ni siquiera luz ni su reflejo (resplandor).
- **Afasia:** es el trastorno del lenguaje que se produce como consecuencia de una lesión o daño cerebral.
  Se trata de la pérdida de capacidad de producir o comprender el lenguaje, debido a lesiones en áreas cerebrales especializadas en estas funciones. Puede ser un trastorno durante la adquisición del lenguaje en los niños o una pérdida adquirida en los adultos. Se relaciona exclusivamente con el lenguaje oral.
  El término afasia, que fue creado en 1864 por el médico francés Armand Trousseau (1801 - 1867), procede del vocablo griego ἀφασία 'imposibilidad de hablar'.
- **Disfasia:** es un trastorno del lenguaje equivalente a una forma menor o indeterminada de afasia.
- **Autismo:** El autismo es un espectro de trastornos caracterizados por un grave déficit del desarrollo, permanente y profundo. Afecta la socialización, comunicación, imaginación, planificación y reciprocidad emocional, y se evidencia mediante conductas repetitivas o inusuales. Los síntomas son la falta de interacción social (muestran dificultad para relacionarse con otros niños de la misma edad, poco o nulo contacto visual, evitan el contacto físico, no responden al ser llamados por su nombre, no tienen lenguaje y si lo tienen presenta alteraciones), las estereotipias (movimientos repetitivos), poca tolerancia a la frustración, risas o llantos sin motivo aparente, presentan hiperactividad o son muy pasivos, no hay juego simbólico, carecen de juego creativo. La mayoría de estos síntomas pueden aparecer al año y medio de edad, comenzando con retrocesos en el desarrollo del niño.
- **Síndrome de Asperger:** es un conjunto de problemas mentales y conductuales que forma parte de los trastornos del espectro autista. Se encuadra dentro de los trastornos generalizados del desarrollo. La persona afectada muestra dificultades en la interacción

social y en la comunicación de gravedad variable, así como actividades e intereses en áreas que suelen ser muy restringidas y en muchos casos estereotípicas.

Se diferencia del autismo infantil temprano y de otras formas menos específicas en que en el trastorno de Asperger no se observa retraso en el desarrollo del lenguaje, y no existe una perturbación clínicamente significativa en su adquisición. No hay retardo, por ejemplo en la edad en que aparecen las primeras palabras y frases, aunque pueden existir particularidades cualitativas (por ejemplo gramaticales) que llamen la atención, así como una preservación generalizada de la inteligencia. Aunque la edad de aparición y detección más frecuente se sitúa en la infancia temprana, muchas de las características del trastorno se hacen notorias en fases más tardías del desarrollo, cuando las habilidades de contacto social comienzan a desempeñar un papel más central en la vida de la persona.

- **Síndrome de Rett:** es una enfermedad congénita con compromiso neurológico que afecta la gran mayoría de las veces al sexo femenino.

  La enfermedad que no es evidente en el momento del nacimiento, se manifiesta generalmente durante el segundo año de vida, y en todos los casos antes de los 4 años. Afecta aproximadamente a 1 niña de cada 10.000. Puede observarse retraso grave en la adquisición del lenguaje y de la coordinación motriz, así como retraso mental grave o severo. La pérdida de las capacidades es por lo general persistente y progresiva.

  El síndrome de Rett provoca grave discapacidad en todos los niveles, haciendo al enfermo dependiente de los demás para el resto de la vida. Toma su nombre del médico austríaco Andreas Rett, que fue el primero en describir la enfermedad en 1966.

- **Trastorno desintegrativo infantil:** es una enfermedad rara caracterizada por una aparición tardía (>2 años de edad) de retrasos en el desarrollo del lenguaje, la función social y las habilidades motrices. Los investigadores no han tenido éxito al encontrar una causa para este desorden.

  Está incluido dentro del trastorno del espectro autista junto con el autismo clásico, síndrome de asperger y autismo atípico pero se suele observar un periodo aparente de desarrollo bastante normal antes de aparecer una regresión (o serie de regresiones) en las habilidades. Muchos niños ya se encuentran con algo de retraso cuando la enfermedad se hace patente, pero estas demoras no son siempre obvias en los niños más pequeños.

  La edad en la cual sobreviene esta regresión varía, y puede ocurrir entre los 2 y los 10 años.

  La regresión puede ser muy repentina, y el niño puede expresar incluso su preocupación sobre lo que está sucediendo, para sorpresa de los padres. Algunos niños describen o parecen estar reaccionando ante alucinaciones, pero el síntoma más obvio es que las habilidades adquiridas aparentemente se pierden. Muchos escritores la han definido como una enfermedad devastadora, que afecta tanto a la familia como al futuro del individuo. Como es el caso de todas las categorías de trastornos generalizados del desarrollo, existe una controversia considerable acerca del tratamiento correcto para este síndrome.

  Fue descrito por el educador austríaco Theodore Heller en 1908, 35 años antes de que Leo Kanner describiera el autismo, pero no se ha reconocido oficialmente hasta hace poco. Heller utilizaba la denominación de “dementia infantilis” para este síndrome.

- **Trastorno del desarrollo no especificado:** puede ser confundido con un tipo específico de autismo “Autismo Atípico”. Sin embargo, este trastorno presenta algunos criterios del resto de los otros Trastornos Generalizados del Desarrollo, pero no cumple

todos los patrones conductuales para determinar que es un Trastorno Autista, Síndrome de Rett, Trastorno Desintegrativo Infantil o Síndrome de Asperger, por lo que es considerado no especificado. Este trastorno afecta a las tres Áreas del Desarrollo: interacción social, comunicación y conducta. No obstante, no manifiesta los comportamientos en todos los trastornos descritos del trastorno generalizado del desarrollo, solo se presentan rasgos de cada uno. A consecuencia de esto, no se conocen sus causas ni su tipo de tratamiento, por tanto, la escasez de información sobre el mismo ha hecho denominarlo "Trastorno Generalizado del Desarrollo no especificado" o "Trastornos Residuales".

5.4 Sistemas Aumentativos y Alternativos de comunicación (SAAC). ARASAAC.

Los Sistemas Aumentativos y Alternativos de Comunicación (SAAC) son formas de expresión distintas al lenguaje hablado, que tienen como objetivo aumentar (aumentativos) y/o compensar (alternativos) las dificultades de comunicación y lenguaje de muchas personas con discapacidad.

La comunicación y el lenguaje son esenciales para todo ser humano, para relacionarse con los demás, para aprender, para disfrutar y para participar en la sociedad y hoy en día, gracias a estos sistemas, no deben verse frenados a causa de las dificultades en el lenguaje oral. Por esta razón, todas las personas, ya sean niños, jóvenes, adultos o ancianos, que por cualquier causa no han adquirido o han perdido un nivel de habla suficiente para comunicarse de forma satisfactoria, necesitan usar un SAAC.

Entre las causas que pueden hacer necesario el uso de un SAAC encontramos la parálisis cerebral (PC), la discapacidad intelectual, los trastornos del espectro autista (TEA), las enfermedades neurológicas tales como la esclerosis lateral amiotrófica (ELA), la esclerosis múltiple (EM) o el párkinson, las distrofias musculares, los traumatismos cráneo-encefálicos, las afasias o las pluridiscapacidades de tipologías diversas, entre muchas otras.

La Comunicación Aumentativa y Alternativa (CAA) no es incompatible sino complementaria a la rehabilitación del habla natural, y además puede ayudar al éxito de la misma cuando éste es posible. No debe pues dudarse en introducirla a edades tempranas, tan pronto como se observan dificultades en el desarrollo del lenguaje oral, o poco después de que cualquier accidente o enfermedad haya provocado su deterioro. No existe ninguna evidencia de que el uso de CAA inhiba o interfiera en el desarrollo o la recuperación del habla.

5.4.1 Recursos utilizados por los SAAC

La Comunicación Aumentativa y Alternativa incluye diversos sistemas de símbolos, tanto gráficos (fotografías, dibujos, pictogramas, palabras o letras) como gestuales (mímica, gestos o signos manuales) y, en el caso de los primeros, requiere también el uso de productos de apoyo. Los diversos sistemas de símbolos se adaptan a las necesidades de personas con edades y habilidades motrices, cognitivas y lingüísticas muy dispares.

Los productos de apoyo para la comunicación incluyen recursos tecnológicos, como los comunicadores de habla artificial o los ordenadores personales y tablets con programas especiales, que permiten diferentes formas de acceso adaptadas algunas para personas con

movilidad muy reducida, y facilitan también la incorporación de los diferentes sistemas de signos pictográficos y ortográficos, así como diferentes formas de salida incluyendo la salida de voz. También pueden consistir en recursos no tecnológicos, como los tableros y los libros de comunicación.

Para acceder a los ordenadores, comunicadores, tableros o libros de comunicación existen diversas estrategias e instrumentos denominados genéricamente estrategias y productos de apoyo para el acceso, tales como los punteros, los teclados y ratones adaptados o virtuales o los conmutadores.

5.4.2 Sistemas de símbolos

En líneas anteriores hemos dividido los sistemas de símbolos para la CAA en gestuales y gráficos. En ambos casos encontramos una gradación desde sistemas muy sencillos, que se adaptan a personas con déficits cognitivos y lingüísticos de diversa consideración, hasta sistemas complejos que permiten niveles avanzados de lenguaje signado (basado en signos manuales) o asistido (basado en signos gráficos).

En el caso de los símbolos gestuales, esta gradación abarca desde el uso de mímica y gestos de uso común hasta el uso de signos manuales, generalmente en el orden correspondiente al lenguaje hablado; es lo que se denomina lenguaje signado o bimodal. Las lenguas de signos utilizadas por las personas no oyentes no se consideran SAAC, ya que constituyen idiomas que se han desarrollado y se adquieren de forma natural, al igual que ocurre con el lenguaje hablado. El uso de signos manuales requiere disponer de habilidades motrices suficientes, como puede ser el caso de personas con discapacidad intelectual o TEA.

Los **símbolos gráficos** abarcan desde sistemas muy sencillos basados en dibujos o fotografías hasta **sistemas progresivamente más complejos** como los **sistemas pictográficos o la ortografía tradicional (letras, palabras y frases).** Gracias a los productos de apoyo para la comunicación y los diversos recursos para el acceso, los sistemas gráficos pueden ser usados por personas con movilidad reducida, incluso en casos de extrema gravedad. Por ello, además de ser usados, como en el caso anterior, por personas con discapacidad intelectual o TEA, los usan también personas con discapacidades motoras (PC, ELA, EM, etc.).

Los sistemas pictográficos se aplican a personas que no están alfabetizadas a causa de la edad o la discapacidad. Tienen la ventaja de permitir desde un nivel de comunicación muy básico, que se adapta a personas con niveles cognitivos bajos o en etapas muy iniciales, hasta un nivel de comunicación muy rico y avanzado, aunque nunca tan completo y flexible como el que se puede alcanzar con el uso de la lengua escrita. Los **sistemas pictográficos más usados** en los diversos territorios del estado español son el sistema SPC (Sistema Pictográfico de Comunicación) y el **sistema ARASAAC**, desarrollado por el Portal Aragonés de CAA y que es de libre disposición con licencia Creative Commons.

5.4.3 Productos de apoyo a la comunicación

Podemos dividir los productos de apoyo para la comunicación en básicos y tecnológicos. Los tableros de comunicación son productos de apoyo básicos que consisten en superficies de materiales diversos en las que se disponen los símbolos gráficos para la comunicación

(fotografías, pictogramas, letras, palabras y/o frases) que la persona indicará para comunicarse. Cuando los símbolos se distribuyen en varias páginas hablamos de libros de comunicación.

Entre los productos tecnológicos encontramos los **comunicadores electrónicos** especialmente diseñados para tal fin y los ordenadores portátiles o las **tablets** con programas especiales que los convierten en comunicadores. Los comunicadores electrónicos dedicados o emulados en ordenadores se personalizan con los símbolos gráficos que requiere cada persona y se caracterizan por ser portátiles y adaptarse a las formas de acceso apropiadas para cada persona (teclados, ratones, conmutadores, etc.). Disponen de una salida para los mensajes en forma de habla digitalizada o sintetizada, así como también, a menudo, de otras salidas como pantalla, papel impreso o incluso funciones de control del entorno.

5.4.4 Fomento del éxito de la intervención con SAAC

Los sistemas y productos de apoyo para la CAA son solamente un medio o una condición necesaria para que la persona con discapacidad del habla pueda comunicarse, desarrollar sus capacidades y participar en el mundo que la rodea, pero no resultan nunca suficientes. Lo verdaderamente importante es el proceso de educación, habilitación y asesoramiento que debe acompañarlos.

El proceso de intervención debe empezar por una evaluación de las capacidades, habilidades, necesidades y deseos de la persona, así como de las características, apoyos, demandas y restricciones de su entorno, con el fin de definir los componentes que va a tener el sistema o sistemas que vayan a resultar más adecuados. Habrá que seleccionar con mucho esmero los productos de apoyo así como las estrategias de acceso y, para usuarios de SAAC no lectores, habrá que realizar una buena selección del vocabulario signado o pictográfico que se va a ir enseñando. Este proceso de evaluación no ha de ser puntual sino continuado a lo largo de la vida.

La habilitación y la enseñanza deben dirigirse tanto a la persona como a su entorno, incluyendo todos los contextos en los que participa o desea participar, así como todas las personas significativas de estos contextos, incluyendo profesionales y, sobre todo, familiares, compañeros y amigos. Esta enseñanza debe llevarse a cabo en entornos educativos y terapéuticos pero también en entornos naturales, en un enfoque de 24 horas que garantice que la persona se verá inmersa en un buen ambiente de lenguaje, rodeada de interlocutores sensibles y competentes, e implicada en actividades interesantes y enriquecedoras.

Para fomentar el éxito de la intervención con SAAC lo más importante es conseguir que la persona con discapacidad de habla tenga cosas interesantes para comunicar a los demás, sepa cómo hacerlo y cuente con interlocutores que quieran escucharle y sepan entenderle. Este objetivo no debe dejarse en manos del azar sino que se debe conseguir a través del esfuerzo y el acierto de profesionales competentes, apoyados por una sociedad cada vez más concienciada y libre de prejuicios.

6. Desarrollo de la aplicación

6.1 Antecedentes

La idea de desarrollar una aplicación como ésta surge de un encuentro con Logopedas del Colegio Público de Educación Especial Alfonso X el sabio de Leganés. En esta reunión se habla del método de trabajo tradicional para la enseñanza de la lectoescritura para niños con determinadas discapacidades y se ponen de manifiesto las ventajas que podrían suponer trasladar algunas de dichas metodologías al mundo multimedia e interactivo. Se identifican las tablets como instrumento ideal para materializar esa nueva forma de trabajo ya que reúnen los requisitos adecuados, como son un tamaño de pantalla suficientemente grande, pantallas táctiles que facilitan la interacción con los dispositivos, y sobre todo la capacidad de reproducir contenidos multimedia de calidad. Además las tablets se han convertido en un elemento que no suele faltar en centros educativos ni en los entornos familiares.

Cuando se habla de dispositivos tipo tablets, nos encontramos tres competidores principales, los basados en Android, los basados en iOS de Apple y los basados en Windows de Microsoft. Se opta por la primera opción por contar con la mayor cuota de mercado[6] según se puede ver en la figura 15.

Table 1: Worldwide Tablet Sales to End Users by Operating System, 2013 (Units)

Operating System	2013 Sales	2013 Market Share (%)	2012 Sales	2012 Market Share (%)
Android	120,961,445	61.9	53,341,250	45.8
iOS	70,400,159	36.0	61,465,632	52.8
Microsoft	4,031,802	2.1	1,162,435	1.0
Others	41,598	<0.1	379,000	0.3
Total	**195,435,004**	**100.0**	**116,348,317**	**100.0**

Source: Gartner (February 2014)

Figura 15. Cuota de mercado de dispositivos tablet a nivel mundial

Se descartan posibles dispositivos como ordenadores, con interfaces de mayor dificultad de manejo (teclados y ratones físicos), o smartphones, cuyas pantallas resultan ser insuficientes para un correcto trabajo con los niños (no olvidemos que algunos pueden presentar discapacidades visuales de diferentes grados y que necesitan tamaños grandes de los diferentes componentes de la aplicación). No obstante la aplicación se ha adaptado también para poder utilizarse en smartphones Android, ya que puede ser útil en determinadas circunstancias, aunque no es el objetivo principal del desarrollo de la aplicación.

Por último, aunque está aplicación está principalmente orientada al trabajo en el colegio, puede ser un instrumento útil para el refuerzo educativo en el seno familiar. Con los métodos tradicionales, se hace uso de fichas físicas con letras, sílabas, pictogramas, etc. Es complicado que las familias puedan contar con estos elementos y más complicado aún es saber cómo realizar las diferentes actividades con ellos. La presenta actividad reúne la disponibilidad de

fichas digitales con otros elementos audiovisuales como locuciones de sílabas y refuerzos sonoros (aplausos en caso de acierto o sonidos de error en caso de fallo), pero lo más importante es que cuenta con una serie de menús que van guiando por las diferentes actividades que comprenden la aplicación. La funcionalidad de dichas actividades está acordada con el equipo de Logopedas del colegio, según su actividad habitual. Por lo tanto guiar a los niños a través de la aplicación es fácil para los profesionales de la educación, pero también para los familiares de los mismos. Se trata pues de una aplicación con actividades tuteladas, donde la persona que acompaña al niño debe indicarle en todo momento el modo de actuar ante cada actividad. No es intención de esta aplicación que el niño trabaje en solitario, aunque tal vez sea posible en algunos casos tras periodo de adaptación y aprendizaje.

6.2 Requisitos de sistema

Como ya se ha mencionado, la aplicación está especialmente orientada a dispositivos Android con pantallas de siete pulgadas o mayores. Dadas las características visuales de la misma, no se recomienda su uso en dispositivos con pantallas menores, si bien su funcionamiento es posible, y dependiendo de las características del mismo el resultado final puede ser más o menos satisfactorio.

La versión mínima de software para que la aplicación pueda ser ejecutada, será Android 3.0 (nivel de API 11). Este requisito es mandatorio, si no se cumple, la aplicación no se ejecutará. Se ha elegido esta versión de partida porque a partir de la misma se introduce el concepto de barra de acciones Android, que sustituye a la anterior barra de título y que aporta nuevas características que son usadas en esta aplicación, como los elementos de navegación.

La cuota de mercado en las versiones Android que superan esta versión supone más del 92% [7], por lo que parece una opción bastante adecuada.

6.3 Entorno de desarrollo

Para el desarrollo de la aplicación, se ha utilizado el entorno de desarrollo integrado (IDE) Eclipse, del que se ha hablado anteriormente, y del que podemos tener una visión general en la figura 16.

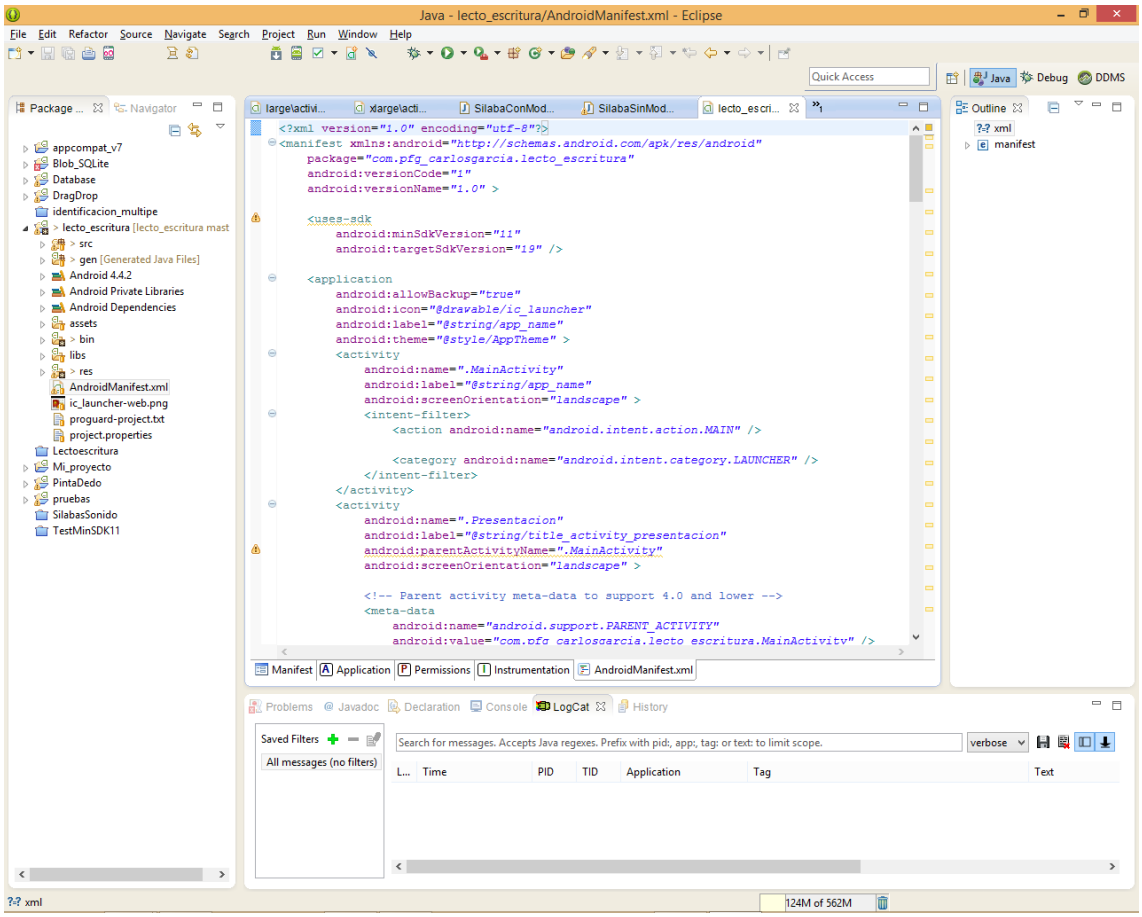


Figura 16. Visión general de Eclipse

Eclipse, como ya se ha dicho, es un entorno de desarrollo integrado, y que reúne en una aplicación todas las funcionalidades necesarias para desarrollar, probar y depurar aplicaciones, y en definitiva todas las herramientas necesarias para gestionar de forma efectiva y sencilla todo el ciclo de vida del desarrollo de aplicaciones como en este caso, en Java para Android.

Existe una alternativa muy potente que está comenzando a ser ampliamente utilizada y que está ganando terreno frente a Eclipse para la programación Android, se trata de Android Studio http://developer.android.com/tools/studio/index.html, pero que en el momento de comenzar con este proyecto aún se encontraba en versión beta, en contraposición con el nivel de madurez de eclipse, y que por tanto, ha sido motivo principal para tomar a la decisión de elegir éste último para el desempeño del presente proyecto.

6.4 Fases del desarrollo

Se tratará de describir brevemente las fases por las que ha pasado este proyecto para intentar ofrecer una mejor visión del desarrollo del mismo:

6.4.1 Toma de contacto

En esta fase podemos englobar las actividades posteriores a la aceptación, por parte de la ETSIST (Escuela Técnica Superior de Ingeniería y Sistemas de Telecomunicación), del PFG (Proyecto de Fin de Grado) propuesto por el autor del mismo. Básicamente se han producido

reuniones con el tutor y con el equipo de logopedia del colegio Alfonso X el Sabio mencionado anteriormente y se ha hecho perfilado a grandes rasgos los resultados que se pretenden conseguir.

6.4.2 Estudio de necesidades

Partiendo del boceto logrado en la toma de contacto, se hace un estudio de los recursos que pueden ser necesarios para el correcto desempeño del proyecto. Se fijan como recurso principal un ordenador personal como herramienta imprescindible para la mayor parte de las tareas necesarias, tanto de búsqueda de información y bibliografía, como herramienta de desarrollo del software y herramientas de pruebas y depuración. No nos podemos olvidar de todas las tareas auxiliares que rodean el desarrollo del proyecto como la redacción de este libro, edición gráfica y de audio, etc.

El segundo recurso imprescindible es una tablet "física" para poder hacer pruebas y depuración en condiciones de un entorno real, como el que se utilizará una vez que la aplicación se encuentre en producción. Como alternativa a las tablets "físicas", nos encontramos con las tablets "virtuales", que nos permiten trabajar cómodamente y con cualquier configuración deseada para realizar la mayor parte del trabajo.

No se ve necesario el uso de otros recursos hardware en forma de dispositivos adicionales de cualquier tipo.

Por último se hace evidente que la lógica de la aplicación necesitará contenidos que gestionar. Esos contenidos serán consensuados con el colegio, atendiendo a sus criterios profesionales en materia de lectoescritura y discapacidad. Los contenidos serán básicamente las sílabas, palabras y pictogramas necesarios para el correcto funcionamiento de la aplicación. Se puede consultar un listado de sílabas y algunas muestras de pictogramas en los anexos.

Por todo lo dicho, no se hace necesario contar con un presupuesto para este proyecto, ya que el autor cuenta con los recursos materiales mencionados necesarios para el desarrollo integral del proyecto.

6.4.3 Análisis de viabilidad de las soluciones previstas

Una vez definidas las especificaciones iniciales previstas en la fase de toma de contacto, se procede a realizar un análisis de viabilidad de las principales actividades. Se procede a realizar un estudio general de la plataforma de desarrollo Android, centrado principalmente en la gestión de vistas ("Views") que serán los que den forma a los componentes visuales de las aplicaciones, en forma de botones ("Buttons", "ImageViews", "EditText", "Layouts" y otros, que serán la base de la parte visual de las diferentes actividades. Así mismo se realiza un estudio sobre la gestión de contenidos de audio y bases de datos. Todos ellos elementos que se prevén imprescindibles en el desarrollo de la aplicación. Tras este estudio, se llega a la conclusión de que la plataforma Android implementa soluciones eficaces y flexibles para llevar a cabo con éxito el desarrollo de la aplicación con éxito. Además, al utilizar el lenguaje de programación Java, se asegura una inmensa bibliografía y foros de consulta para cualquier dificultad encontrada en cualquier momento.

6.4.4 Desarrollos y pruebas por actividades individuales

Se ha seguido una filosofía de trabajo por estructuras, es decir, se ha creado primero la actividad principal donde se eligen las sílabas de trabajo. Una vez elegidas dichas sílabas se pasa al menú principal donde se debe elegir que grupo de actividades se quiere trabajar. Se ha creado esa estructura de menú y se ha dotado de la lógica necesaria para realizar saltos a cada submenú, uno por cada grupo de actividades. Posteriormente se han creado estas últimas estructuras de menú y se ha creado la estructura de navegación por la aplicación en base a flechas de actividad anterior y posterior, y un menú adicional de selección de grupos en la barra de acciones de Android. Hasta este punto no se ha dotado de lógica a las actividades, sólo se han creado con el código por defecto (el típico mensaje "Hello World!" tan conocido en el mundo de la programación).

Posteriormente, se dota de código funcional a cada una de las actividades, siguiendo el orden de los menús, para llevar un mejor control de los pasos ya realizados en la implementación, aunque se podría seguir cualquier orden, ya que cada actividad es independiente de las demás en cuanto a su codificación (existe algún nexo común, como es el caso de métodos auxiliares, de los que se hablará más adelante).

En aras de la calidad y de seguir con el orden establecido de programación de la mejor manera posible, se ha decidido realizar pruebas de funcionamiento y depuración inmediatamente después de la finalización de la programación de cada actividad, e incluso, en numerosas ocasiones, antes de terminar la actividad, para probar piezas parciales del código, cuando se ha considerado conveniente. Tal vez, una marcha atrás en esas partes podría suponer una cantidad de trabajo considerable para rehacer el trabajo. Parece más adecuado pues, probar una actividad cuando se tiene reciente el proceso de programación que dejarlo como última acción cuando ya se han programado todas las actividades y se puede haber olvidado en parte los entresijos de dicha actividad (no olvidemos que contamos con más de veinte actividades en la aplicación).

6.4.5 Retoques estéticos.

Se ha dejado para el final del proyecto abordar la estética de la aplicación. Desde un principio se ha dotado a la aplicación de elementos visuales básicos (hablamos de la distribución de botones y su estética propia, así como del icono fundamentalmente), y no se han tenido en cuenta otros recursos estéticos como colores de fondo. Ha sido entonces, cuando se ha comprobado la correcta funcionalidad de la aplicación, cuando se ha comenzado a trabajar en estos aspectos.

Se han tenido en cuenta las observaciones de los profesionales logopedas en este respecto, ya que los elementos visuales recargados pueden resultar contraproducentes para el objetivo perseguido, que es la enseñanza de la lectoescritura, de forma lo más amena posible, pero sin llegar a despistar al niño del trabajo a realizar. Básicamente podemos resumir la estética como de carácter austero y con colores suaves, prestando mayor énfasis en el uso de pictogramas adecuados para el desempeño de las actividades.

6.5 Descripción de la aplicación

Con esta aplicación se persigue facilitar la enseñanza de la lectoescritura (enseñanza y aprendizaje de la lectura simultáneamente con la escritura[8]) orientada a niños con necesidades

educativas especiales. La unidad básica de trabajo será la sílaba y en torno a cada sílaba se irán realizando distintas actividades para facilitar el conocimiento de dicha sílaba.

Conviene dejar claro que toda la aplicación está centrada en el uso exclusivo de letras mayúsculas. Esto es así porque la iniciación a la lectoescritura se desarrolla de esta manera por motivos pedagógicos. No podemos olvidar que esta aplicación se enfoca dentro del inicio del aprendizaje de la lectoescritura.

Pasamos a describir las diferentes partes en que se compone la aplicación.

6.5.1 Actividad principal o pantalla de inicio

La elección de la sílaba de trabajo se realizará desde la pantalla principal ("main activity" en términos de programación Android, figura 17). En dicho panel se presentan en forma de botones todas las opciones que tenemos para la elección de las sílabas de trabajo. Podemos diferenciar tres tipos de botón. Por un lado tenemos los botones que representa la primera letra o grupo de letras que formarán las distintas sílabas de forma regular, es decir, cinco sílabas formadas con el contenido del botón más cada una de las cinco vocales (**"P"** para formar **"PA, PE, PI, PO, PU"**, **"FR"** para formar **"FRA, FRE, FRI, FRO, FRU"**). Por otro lado tenemos los botones que presentan algún tipo de irregularidad y que llevan a trabajar únicamente las sílabas representadas dentro del botón (en algunos cosas dos sílabas y en otros tres):

- **"CA, CO, CU"**
- **"QUE, QUI"**
- **"GA, GO, GU"**
- **"GUE, GUI"**
- **"GE, GI"**
- **"ZA, ZO, ZU"**
- **"CE, CI"**

Por último nos encontramos con la peculiaridad de las sílabas formadas con la **"R suave"** ("MU**RO**"), en contraste con la **"R"** fuerte ("**RO**MA"). Ambas se escriben igual, pero tienen diferente sonido. La convención usada para diferenciar la suave de la fuerte es anteponer un guion delante de la sílaba suave: **"-RO"**. En las actividades donde se represente la sílaba de forma aislada se hará uso del guion, no así cuando forme parte de una palabra completa.

Estas irregularidades conllevan una serie de controles y otras actuaciones en la codificación de la aplicación que ha supuesto un incremento en la complejidad del desarrollo del conjunto.

Como se puede observar, la forma de ordenar las sílabas en la pantalla principal no corresponde con el orden alfabético, sino con un orden establecido con criterios didácticos por el equipo de logopedas que han colaborado en los requisitos de este proyecto.

Figura 17. Panel para elección de la sílaba de trabajo

6.5.2 Grupos de actividades

Contamos con cinco grupos de actividades (Presentación, Discriminación Visual, Discriminación Auditiva, Lectura, Escritura, como se puede ver en la figura 18). Dentro de cada grupo se presentarán distintas actividades que permitirán el trabajo objetivo del grupo desde diferentes puntos de vista, o simplemente alternando el uso de colores en las sílabas o el uso solo del color negro. Respecto al color de las sílabas se ha seguido un código de colores fijo para todas las actividades. Las sílabas que se forman con la vocal "A" serán de color rojo, el azul será para la "E", amarillo para la "I", verde para la "O" y naranja para la "U". Podemos ver un ejemplo en la figura 19 (actividad "Presentación de sílabas color" dentro del grupo "Presentación").

Figura 18. Grupos de actividades

Figura 19. Actividad Presentación de sílabas color

En contraposición tenemos, en la figura 20, la misma actividad pero todas las sílabas en color negro.

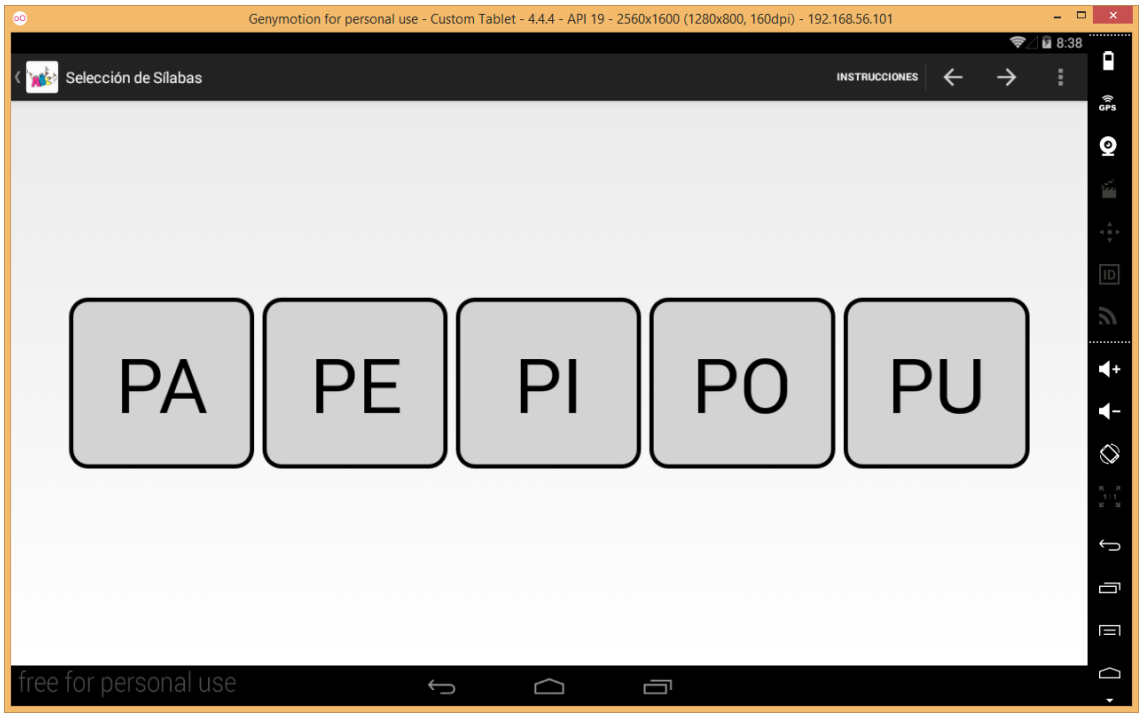


Figura 20. Actividad Presentación de sílabas negro

Esta diferenciación se hace para facilitar el reconocimiento de las sílabas en el caso del color, y para añadir un grado de dificultad en la actividad, en el caso del negro, cuando ya se han obtenido progresos con las sílabas en color.

6.5.3 Navegación por la aplicación. Instrucciones

Antes de pasar a describir cada una de las actividades, conviene hablar de la navegación para moverse entre dichas actividades. El siguiente sería el esquema de actividades de la aplicación:

- **PRESENTACIÓN**
  - Sílabas ordenadas color
  - Sílabas desordenadas color
  - Sílabas ordenadas negro
  - Sílabas desordenadas negro

- **DISCRIMINACIÓN VISUAL**
  - Identificación sílabas color
  - Identificación sílabas negro
  - Identificación múltiple color
  - Identificación múltiple negro
  - Identificación parejas color
  - Identificación parejas negro
  - Identificación en palabra color
  - Identificación en palabra negro

- **DISCRIMINACIÓN AUDITIVA**
  - Discriminación sílabas color
  - Discriminación sílabas negro
  - Discriminación sílabas color en palabra
  - Discriminación sílabas negro en palabra
- **LECTURA**
  - Asociación palabra pictograma
- **ESCRITURA**
  - Pinta la sílaba
  - Escribe la sílaba con modelo
  - Escribe la sílaba sin modelo
  - Escribe la sílaba en palabra con modelo
  - Escribe la sílaba en palabra sin modelo
  - Formación de palabras

Comenzamos a trabajar seleccionando las sílabas de trabajo desde la pantalla de principal (figura 17). Esto nos llevará al menú para elegir el grupo de actividades (figura 18). Aquí será la persona que esté guiando al niño en la actividad quien tome la decisión de elegir el grupo de actividades que corresponda.

El uso lógico por motivos pedagógicos de la aplicación sería en primer lugar elegir las sílabas de trabajo, comenzando por el primer botón (de arriba hacia abajo y de izquierda a derecha, es decir comenzar por las sílabas "PA, PE, PI, PO, PU"), y una vez realizadas todas las actividades de todos los grupos referidas a dichas sílabas, continuar con el siguiente botón ("LA, LE, LI, LO, LU") y con sus actividades correspondientes. El orden de trabajo de grupos de actividades se debe realizar siguiendo el orden en que está estructurado el menú de la figura 18, desde arriba hacia abajo y de izquierda a derecha, criterio que se usará de igual modo para las actividades dentro de cada grupo.

En resumen, habría que seguir el esquema de actividades citado un poco más arriba en orden descendente, trabajando las sílabas en el orden también citado en el párrafo anterior.

Para movernos por la aplicación, entre actividades existen dos opciones. La primera es una vez seleccionadas las sílabas de trabajo, ir accediendo a los distintos menús presentes en la aplicación. La otra es comenzar con una actividad y una vez finalizada pasar a la siguiente o a la anterior haciendo uso de las flechas de navegación adelante y atrás de la barra de acciones ("Action Bar" en inglés), como vemos en la figura 21.

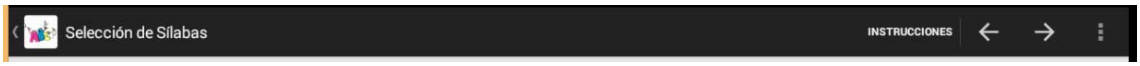


Figura 21. Barra de acciones

Con estas flechas pasamos de actividad sin tener en cuenta si se cambia el grupo de actividades o no. Por ejemplo, cuando estamos en la última actividad del grupo "Presentación", "Sílabas desordenadas negro", y pulsamos sobre la flecha hacia la derecha (siguiente actividad), pasaremos directamente a la primera actividad de "Discriminación visual", es decir, a la actividad "Identificación sílabas color". En todo caso se van arrastrando las sílabas de trabajo

actuales entre actividades. Casos particulares de navegación son la primera actividad "Sílabas ordenadas color", que solo presenta la flecha de navegación de actividad siguiente, y la última actividad, "Formación de palabras", que solo presenta la flecha de navegación de actividad anterior. De este modo se impide un comportamiento circular de navegación, especificando con claridad cuál es el comienzo y cuál es el fin de las actividades de unas sílabas de trabajo.

Para facilitar la compresión de los objetivos de las diferentes partes de la aplicación, se ha incluido en la barra de acciones, un botón "INSTRUCCIONES" (ver figura 21), que una vez pulsado presentará un mensaje de texto ("Toast") en pantalla con una breve explicación de la actividad y simultáneamente se reproducirá una locución con el mismo mensaje.

Dentro de la barra de acciones encontramos también el icono de la aplicación a la izquierda de la misma (figura 21). En cualquier momento y en cualquier actividad si pulsamos sobre el icono, nos moveremos a la pantalla principal y podremos cambiar las sílabas de trabajo.

Por último, en el extremo derecho de la barra (figura 21), podemos ver tres puntos alineados en vertical. Pulsando aquí tendremos acceso a un menú donde podremos elegir el grupo de actividades de los cinco posibles y que ya se han mencionado anteriormente (figura 22).

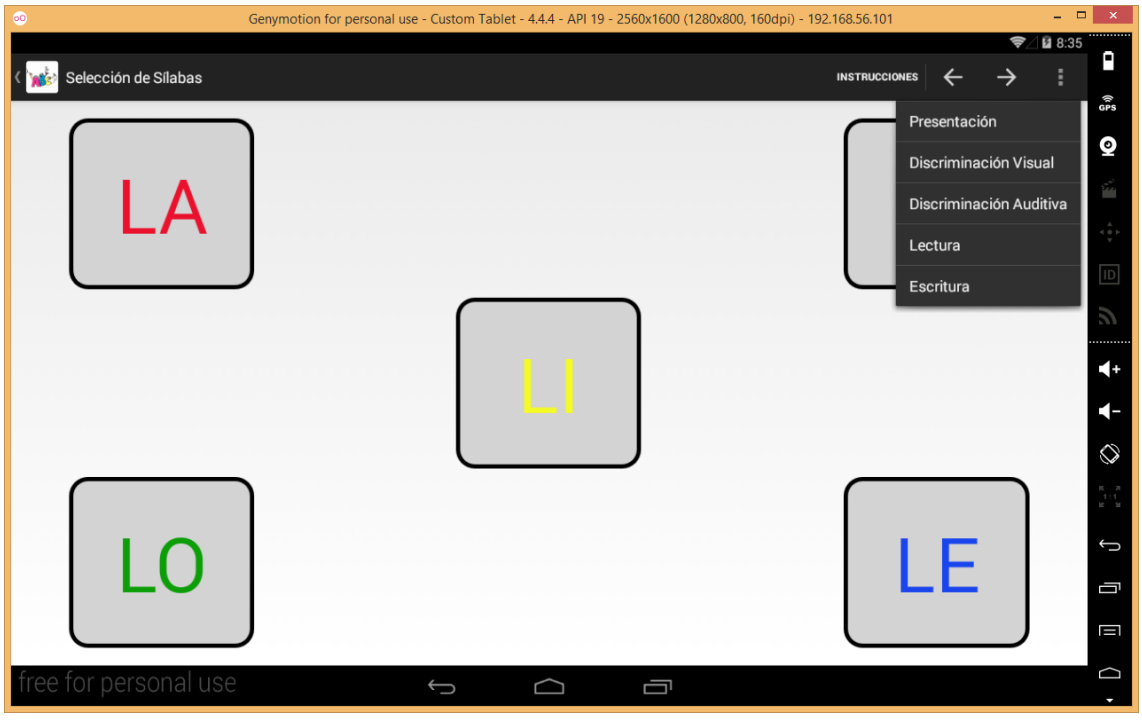


Figura 22. Menú de grupos en barra de acciones

Contamos además con otras opciones de navegación que son las propias de la barra de navegación del sistema Android, que podemos ver en la figura 23.

23. Barra de navegación de Android

Esta barra nos ofrece la posibilidad de volver a la actividad anterior (flecha hacia la izquierda), acceder directamente al escritorio de Android, o comprobar las aplicaciones abiertas.

Veamos ahora una breve descripción de cada grupo y actividad.

6.5.4 Presentación

Con las actividades de este grupo (figura 24) se persigue que los niños hagan una toma de contacto con el mundo de la lectoescritura, presentando las diferentes sílabas para su reconocimiento visual y se añade el refuerzo sonoro por medio de las locuciones correspondientes al sonido de cada una de las sílabas. Estas actividades no tienen un fin de actividad definido, sino que cuando se considere oportuno se deberá pasar a la siguiente actividad haciendo uso de las flechas de navegación de la barra de acciones.

- **Sílabas ordenadas color:** Se presentan todas las sílabas en orden alfabético y el niño tiene que interactuar con ellas pulsado sobre cada botón, obteniendo como resultado la locución correspondiente. Se usa el código de colores establecido.
- **Sílabas desordenadas color:** Se presentan todas las sílabas distribuidas en orden aleatorio por la pantalla, cuidando de mantener una separación adecuada entre ellas, y el niño de nuevo tiene que interactuar con ellas. Se usa el código de colores establecido.
- **Sílabas ordenadas negro:** igual que la actividad sílabas ordenadas color, pero todas las sílabas en negro.
- **Sílabas desordenadas negro:** igual que la actividad sílabas desordenadas color, pero todas las sílabas en negro.

Figura 24. Actividades de "Presentación"

6.5.5 Discriminación Visual

Se trata del grupo con mayor número de actividades (figura 25). El objetivo principal es trabajar el reconocimiento visual de las sílabas de trabajo con las diferentes actividades propuestas. Volvemos a contar con dos versiones de cada actividad, con las sílabas en colores o bien todas las silabas en negro. Para las primeras seis actividades no hay un punto de finalización de actividad definido, en el momento considerado oportuno se debe cambiar de actividad pulsando la flecha de navegación siguiente. En las últimas dos actividades, sin embargo no ocurre lo mismo, es decir, una vez que se ha trabajado cada sílaba una sola vez, se pasa a la actividad siguiente.

- **Identificación sílabas color:** se presenta una sílaba que será el modelo, sobre un botón con el borde azul y después se presenta un grupo de tres sílabas en una columna a la derecha del modelo, o bien tres sílabas en una fila debajo del modelo. Del grupo de tres sílabas una de ellas se corresponde con el modelo, y las otras dos se presentan aleatoriamente del resto de posibles vocales, sin posibilidad de repetición. Se trata de elegir de entre las tres sílabas la que es igual que el modelo, pulsando sobre su botón correspondiente. Se van alternando las dos opciones de visualización (vertical u horizontal) a lo largo de la actividad. Cuando se pulsa sobre una sílaba que no corresponde con el modelo se reproduce una locución de error y se perfila el borde del botón de color rojo. Si el botón pulsado es el correcto, entonces la locución será un aplauso y el botón se perfila de color azul. En todo momento se mantiene el código de colores para las sílabas.
- **Identificación sílabas negro:** igual que la actividad identificación sílabas color, pero todas las sílabas en negro.
- **Identificación múltiple color:** se presenta una sílaba que será el modelo, sobre un botón con el borde azul y después se presenta una serie de botones con varias sílabas. Tres de dichos botones contienen la sílaba correspondiente al modelo y los otros cinco contienen otras sílabas diferentes. El objetivo de la actividad es pulsar sobre los tres botones de la sílaba modelo. Cuando se produce un acierto, el borde del botón se vuelve azul y pasado un segundo desparece de pantalla. Cuando se han producido los tres aciertos, se reproduce un sonido de aplauso y se reinicia la actividad con otra disposición de botones de forma aleatoria y con la siguiente sílaba modelo en orden alfabético. En caso de error al pulsar sobre un botón, se reproduce un sonido de error, y el borde del botón se vuelve rojo durante la locución.
- **Identificación múltiple negro:** igual que la actividad identificación múltiple color, pero todas las sílabas en negro.
- **Identificación parejas color:** se presentan dos filas de botones, cada una con todas las sílabas de trabajo. En ambos casos, las sílabas están ordenadas de forma aleatoria. Se trata de emparejan una sílaba de la fila superior, con su correspondiente en la fila inferior arrastrando desde la primera y soltando en la segunda. En caso de acierto, suena la locución correspondiente a la silaba emparejada y ambas sílabas desaparecen. En caso de error la sílaba de la fila superior vuelve a su posición original y se reproduce un sonido de error. Al emparejar todas las sílabas, se reproduce la locución de aplauso y se reinicia la actividad.
- **Identificación parejas negro:** igual que la actividad identificación parejas color, pero todas las sílabas en negro.
- **Identificación en palabra color:** se presenta una sílaba modelo que será la que habrá que descubrir dentro de una palabra completa. La palabra será de color negro, excepto

la sílaba de trabajo que será de su color correspondiente. Además se presenta un pictograma que representa dicha palabra. La distribución en la pantalla es en la parte superior la sílaba modelo, debajo de ésta, la palabra, y por último en la parte más baja, el pictograma. Tenemos la posibilidad de pulsar sobre las diferentes sílabas que forman la palabra. Cuando se produzca el acierto, se pasa a la siguiente sílaba de trabajo. Una vez concluida la actividad con todas las sílabas de trabajo, se pasa a la siguiente actividad: "Identificación en palabra negro".

- **Identificación en palabra negro:** igual que la actividad identificación en palabra color, pero todas las sílabas en negro. Al finalizar la actividad se pasa a la primera actividad del grupo "Discriminación auditiva", que es "Discriminación sílabas color".

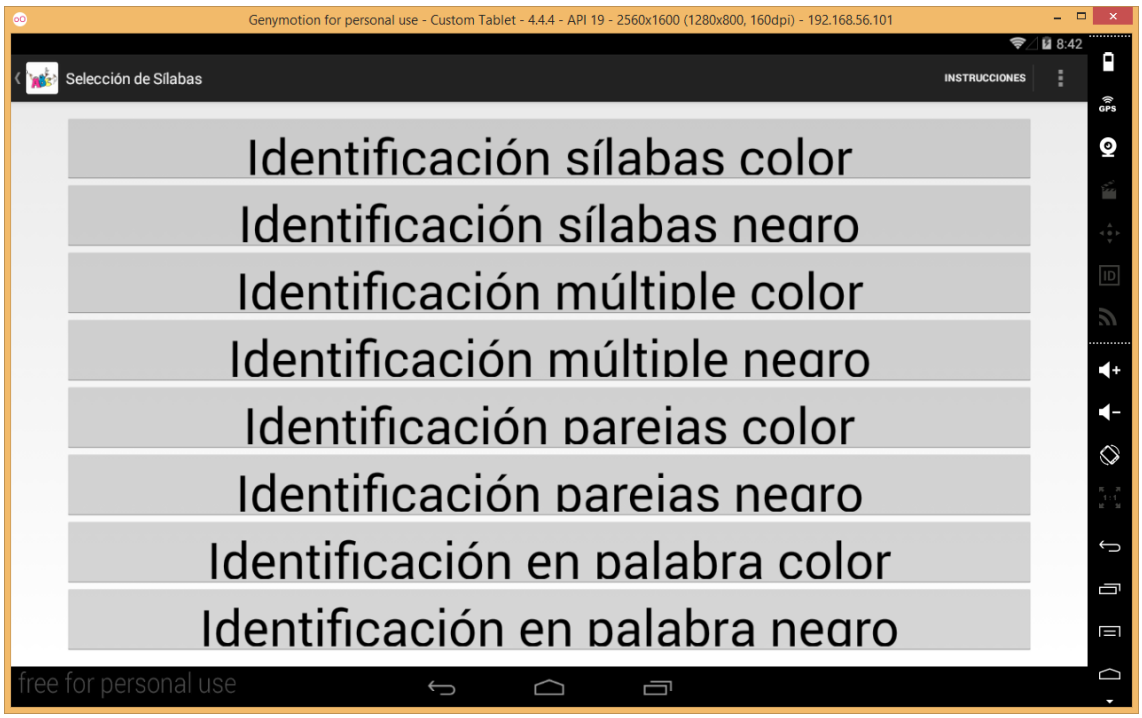


Figura 25. Actividades de "Discriminación visual"

6.5.6 Discriminación Auditiva

El objetivo principal es trabajar el reconocimiento auditivo de las sílabas de trabajo con las diferentes actividades propuestas. Contamos de nuevo con dos versiones de cada actividad, con las sílabas en colores o bien todas las silabas en negro (figura 26). Ahora ya todas las actividades se realizan una única vez con todas las sílabas y una vez finalizadas correctamente, se pasa a la actividad siguiente.

- **Discriminación sílabas color:** Al iniciar la actividad se escucha una locución con instrucciones "Pulsa sobre la sílaba que vas a escuchar". A continuación se escucha el sonido de la sílaba de trabajo y se visualizan tres sílabas y hay que pulsar sobre la que corresponde a la locución. Se producirá los ya conocidos sonidos de error o de aplauso en caso de acierto. Tras un acierto, se pasa a la siguiente sílaba hasta completar la actividad. Las dos sílabas incorrectas son presentadas de forma aleatoria.

- **Discriminación sílabas negro:** igual que la actividad discriminación sílabas color, pero todas las sílabas en negro.
- **Discriminación sílabas color en palabra:** Al iniciar la actividad se escucha una locución con instrucciones "Pulsa sobre la sílaba que vas a escuchar". A continuación se escucha el sonido de la sílaba de trabajo y se presenta un pictograma y debajo la palabra que representa. Dicha palabra contiene la sílaba recién escuchada. El objetivo es pulsar sobre la sílaba correcta, con las habituales opciones de acierto o error vistas hasta ahora.
- **Discriminación sílabas negro en palabra:** igual que la actividad discriminación sílabas color en palabra, pero todas las sílabas en negro.

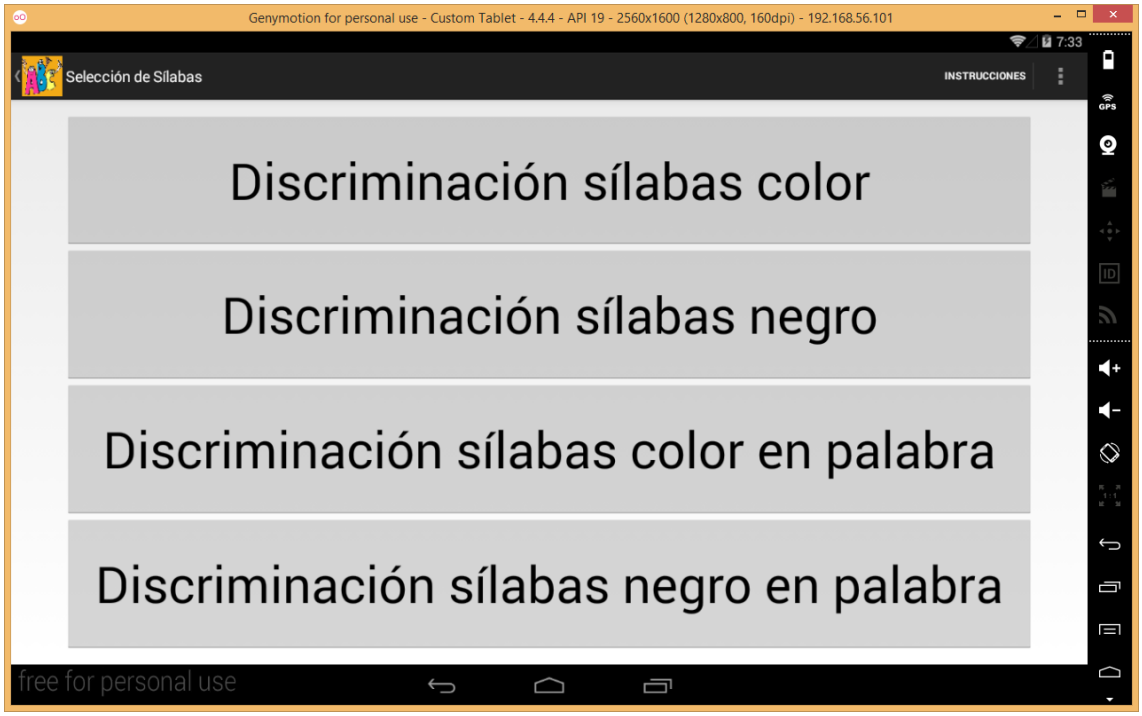


Figura 26. Actividades de "Discriminación visual"

6.5.7 Lectura

En realidad este grupo no lo es tal, ya que solo contiene una actividad, pero se mantiene la categoría de grupo por motivos organizativos. Como se desprende del título, la actividad intenta fomentar las habilidades para la lectura de palabras sencillas (figura 27).

- **Asociación palabra pictograma:** nos encontramos con un pictograma en la pantalla y debajo del mismo tres palabras alineadas en horizontal. Una de las tres se corresponde con el pictograma, y se coloca de forma aleatoria en cualquiera de las tres posibles posiciones. Las dos palabras erróneas tienen alguna similitud con la palabra correcta con la intención de hacer más interesante la actividad. Una vez concluida la ronda de sílabas, se pasa a la siguiente actividad "Pinta la sílaba", que pertenece ya al grupo de "Escritura".

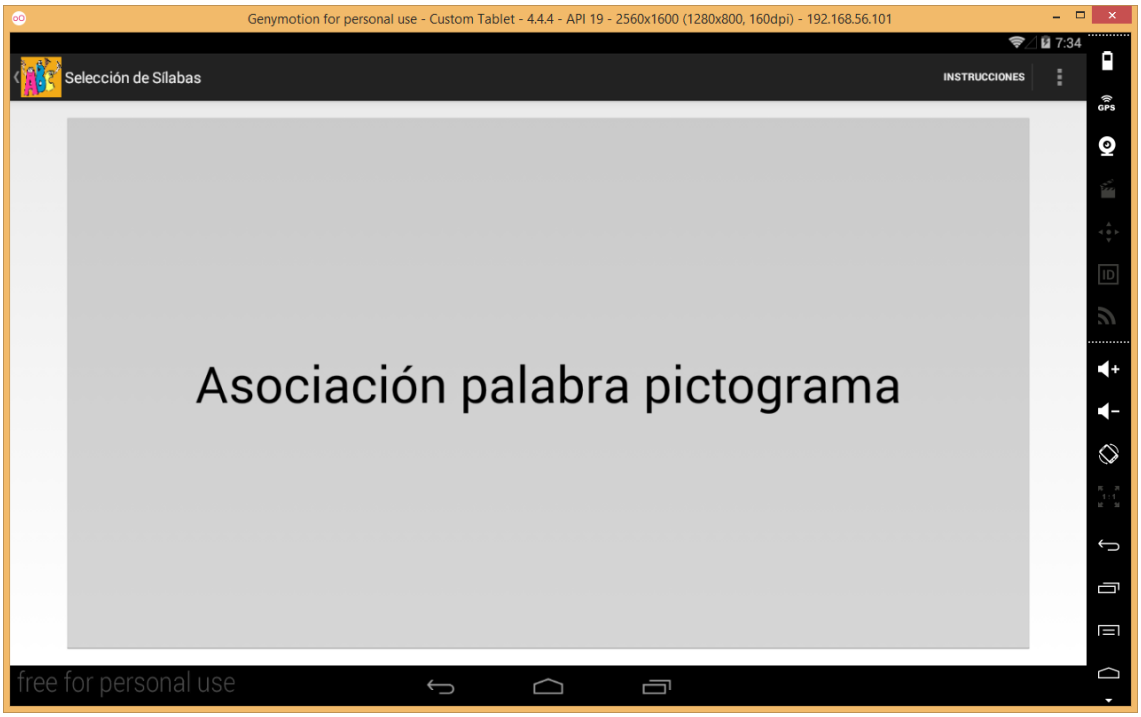


Figura 27. Actividades de "Lectura"

6.5.8 Escritura

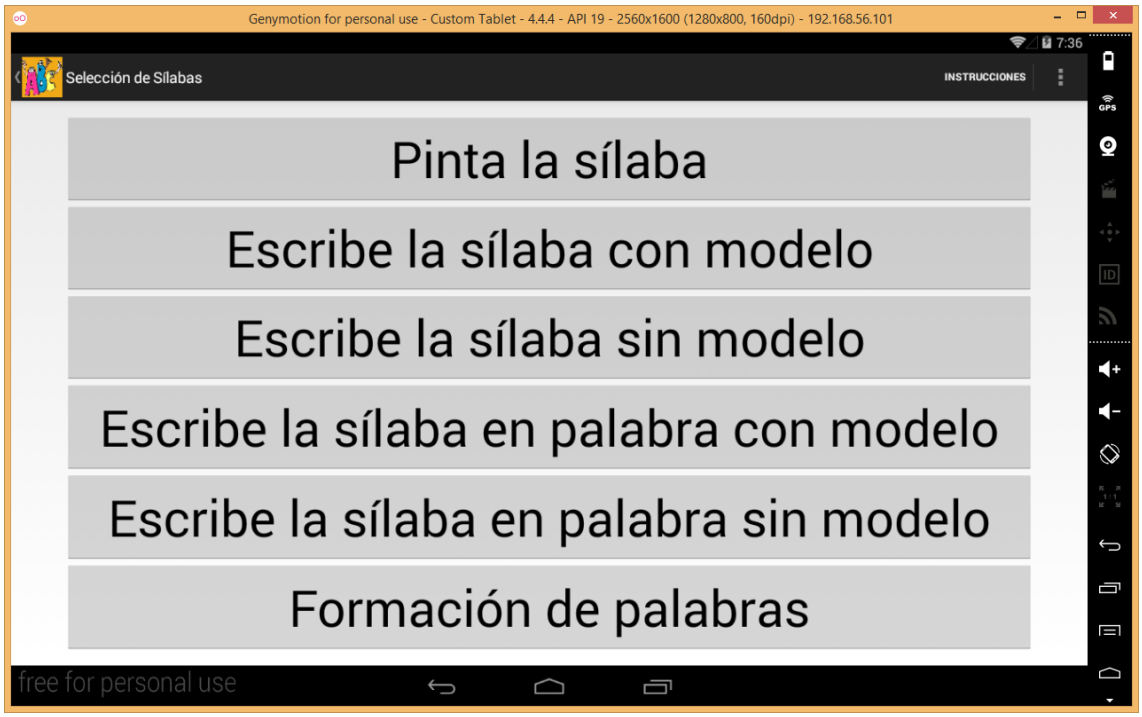
Las últimas actividades se engloban en este grupo (figura 28). El objetivo, como se puede deducir del título es desarrollar las habilidades para la escritura. Se trabajará sobre todo la escritura por medio del teclado virtual de Android, pero también se practicará la habilidad grafomotricidad aprovechando los recursos de pantalla táctil que nos ofrecen las tablets, es decir, realizando trazos con los dedos sobre la pantalla. En estas actividades ya no se hace uso del código de colores para las sílabas, se trabaja directamente en negro, puesto que se debería haber llegado hasta aquí habiendo pasado previamente por todas las actividades anteriores. Esto supone un cierto grado de conocimiento de las sílabas de trabajo y no parece ya necesario facilitar el trabajo mediante el uso de colores.

Con este grupo se cierra una serie de actividades que pretenden hacer un pequeño recorrido por todos los aspectos relacionados con la enseñanza de la lectoescritura orientada a dispositivos móviles. Actividades del grupo "Escritura":

- **Pinta la sílaba:** se presenta la sílaba de trabajo en pantalla, ocupando su totalidad. Las letras son huecas para poder rellenar su interior con el trazo del dedo. Cuentan con una guía indicativa de la dirección del trazado. Esta actividad no tiene validación, es decir, la persona que está junto al niño, cuando considere la actividad realizada, debe pulsar sobre el botón "HECHO", para pasar a la siguiente sílaba. Es previsible que se produzcan errores o descuidos en los trazados, por ello se ha habilitado un botón "BORRAR" que limpia toda la actividad.
- **Escribe la sílaba con modelo:** se presenta una sílaba y a su lado aparece un "Edittext" (zona para poder escribir con el teclado virtual de Android). Aquí habrá que escribir una por una las letras de dicha sílaba, con opción de borrar como cualquier otra operación con el teclado de Android. Una vez realizado, se debe pulsar sobre un botón "HECHO"

para proceder a comprobar si lo que se ha escrito corresponde con la sílaba visualizada. En caso de acierto se reproduce un aplauso y se pasa a la siguiente sílaba.

- **Escribe la sílaba sin modelo:** se escucha una sílaba. En la pantalla aparece un "Edittext" en el que habrá que escribir dicha sílaba. Se ha incluido un botón "ESCUCHAR" por si se quiere volver a escuchar la sílaba. También contamos con el botón "HECHO", para comprobar si lo que se ha escrito corresponde con la sílaba visualizada. En caso de acierto se reproduce un aplauso y se pasa a la siguiente sílaba.
- **Escribe la sílaba en palabra con modelo:** se presenta una palabra y su pictograma correspondiente. A su lado, aparece la misma palabra a la que le falta la sílaba de trabajo, y en su lugar aparece un "Edittext" para poder introducir dicha sílaba. Una vez realizado, se debe pulsar sobre un botón con un "Check" para proceder a comprobar si lo que se ha escrito corresponde con la sílaba que falta. En caso de acierto se pasa a otra sílaba.
- **Escribe la sílaba en palabra sin modelo**: se presenta un pictograma, ésta vez sin que aparezca su palabra correspondiente. A su lado, aparece dicha palabra a la que le falta la sílaba de trabajo, y en su lugar aparece un "Edittext" para poder introducir dicha sílaba. Una vez realizado, se debe pulsar sobre un botón con un "Check" para proceder a comprobar si lo que se ha escrito corresponde con la sílaba que falta. En caso de acierto se pasa a otra sílaba.
- **Formación de palabras:** se presenta un pictograma, ésta vez sin que aparezca ninguna palabra ni otra ayuda que el propio pictograma. A su lado aparece un "Edittext" para poder introducir la palabra correspondiente al pictograma. Una vez escrita la palabra, se debe pulsar sobre un botón con un "Check" para proceder a comprobar si lo que se ha escrito corresponde con la palabra. En caso de acierto se pasa a otra sílaba. Al tratarse de la última actividad, cuando se ha pasado por todas las sílabas de trabajo, la aplicación nos lleva a la pantalla principal para poder elegir otro grupo de sílabas de trabajo.



28. Actividades de "Escritura"

6.5.9 Orden de trabajo de las sílabas

Se ha considerado conveniente que el orden de trabajo de las sílabas dentro de una actividad sea llevado a cabo de forma aleatoria, es decir, que si las sílabas de trabajo son "PA, PE, PI, PO, PU", no sea siempre la sílaba "PA" la que comience una actividad, sino cualquiera de las cinco, lo que se consigue realizando una llamada a un método Java "generaAleatorio(int base)", codificado ad-hoc para implementar este proceso aleatorio . De este modo se evita que los niños aprendan que siempre la primera sílaba de trabajo será la formada con la vocal "A", evitando que se guíen por una sucesión lógica. Esta forma de presentar las sílabas no afecta a algunas actividades en las que carece de sentido, como por ejemplo cuando se presenta todas las sílabas (Actividades de "Presentación").

6.6 Implementación de la aplicación

Como ya se ha comentado, las actividades de Android (en este caso, cada una de las actividades de trabajo de los diferentes menús: "Sílabas ordenadas color", "Identificación múltiple negro", etc.) se componen de un fichero con la clase Java que implementa la funcionalidad de la actividad, y otro fichero que describe en formato XML el "Layout" o distribución de la pantalla o interfaz gráfico correspondiente a dicha actividad. Veamos un ejemplo con extractos de la actividad "Sílabas ordenadas color".

Extracto de Clase “SilabasOrdenadas.java”:

```
protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_silabas_ordenadas);
            setTitle(R.string.seleccion_letra);

            // Obtener mensaje del intent (la consonante o grupo de letras
            // de la silaba y si es en color o negro)
            Intent intent = getIntent();
            letra = intent.getStringExtra("letra_grupo");
            color = intent.getStringExtra("color");

            cntx_silabasordenadas = getApplication();

            // Crear texto de las silabas
            silabas = Metodos_aux.crea_texto_silaba(letra);

            silaba_A = (Button) findViewById(R.id.Silaba_A);
            silaba_E = (Button) findViewById(R.id.Silaba_E);
            silaba_I = (Button) findViewById(R.id.Silaba_I);
            silaba_O = (Button) findViewById(R.id.Silaba_O);
            silaba_U = (Button) findViewById(R.id.Silaba_U);

            switch (letra) {
            // El layout normal tiene cinco botones, pero se diferencia
            // para las silabas que son solo dos o tres, dependiendo de
            //la letra o grupo de letras que se pulsó en la actividad de
            // selección de letra.
            case "CA CO CU":
            case "GA GO GU":
            case "ZA ZO ZU":
                  silaba_E.setVisibility(View.INVISIBLE);
                   //Ocultamos el segundo boton
                  silaba_O.setVisibility(View.INVISIBLE);
                   //Ocultamos el cuarto boton

                  //Sílaba con A

                  //Se asigna la silaba formada con la "A" al boton
                  silaba_A.setText(silabas[0]);
                  //Locución con "A"
                  silaba_A.setOnClickListener(new View.OnClickListener() {
                        @Override
                        public void onClick(View v) {
                        //Obtenemos el id de la locución para
                        //la silaba con "A"
```

Extracto de definición de Layout para la actividad “Silabas ordenadas color” (activity_silabas_ordenadas.xml)

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_weight="1"
    android:gravity="center"
    android:orientation="horizontal" >

    <Button
        android:id="@+id/Silaba_A"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_marginBottom="5dp"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:background="@drawable/forma_boton_presentacion"
        android:text="Silaba_A"
        android:textSize="90sp" />

    <Button
        android:id="@+id/Silaba_E"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_marginBottom="5dp"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:background="@drawable/forma_boton_presentacion"
        android:text="Silaba_E"
        android:textSize="90sp" />

    <Button
        android:id="@+id/Silaba_I"
        android:layout_width="0dp"
        android:layout_height="match_parent"
        android:layout_marginBottom="5dp"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:layout_marginTop="5dp"
        android:layout_weight="1"
        android:background="@drawable/forma_boton_presentacion"
        android:text="Silaba_I"
        android:textSize="90sp" />
```

La aplicación se ha construido en base a las diferentes clases y ficheros XML asociados que constituyen cada actividad. Cuando se ejecuta la aplicación, se invoca a la actividad principal (“MainActivity.java”, “activity_main.xml”). Como ya se ha dicho, esta aplicación presenta una matriz de botones, para seleccionar las sílabas que se pretende comenzar a trabajar. Una vez pulsado el botón deseado, se llama a la actividad de “selección de actividad” (“SeleccionActividad.java”, “activity_seleccion_actividad.xml”), donde aparecerá el menú que nos permite elegir el grupo de actividades de entre los cinco posibles (“Presentación”, “Discriminación visual”, “Discriminación auditiva”, “Lectura” y ”Escritura”).

En dicha llamada a la actividad de “selección de actividad” se incluyen las sílabas de trabajo previamente elegidas. Posteriormente, se irá saltando entre las diferentes actividades que vayamos trabajando y en todo momento se debe transmitir cuáles son esas sílabas de trabajo para mantener un flujo de trabajo lógico. Es decir, según se vaya cambiando de actividad, bien de forma automática pasando a la actividad siguiente por finalización de la actual, bien de forma discrecional, seleccionando una determinada actividad por medio de los diferentes menús de aplicación o botones de navegación, en todo momento se tienen que transferir la información de que sílabas son las correspondientes en un momento dado. La forma de implementar el envío de la información de dichas sílabas de trabajo entre actividades es mediante el uso de los recursos “Intent” que proporciona la infraestructura de Android. Veamos un ejemplo de paso de información de sílabas haciendo uso de “Intents”:

```
      case R.id.action_forward:
            if (color.equals("color"))
            {
                  Intent intent_forward = new
Intent(cntx_silabasordenadas, SilabasDESordenadas.class);
      //Actividad siguiente
                  intent_forward.putExtra("letra_grupo", letra);
                  intent_forward.putExtra("color", "color");
                  startActivity(intent_forward);
                  finish();
            }
            else
            {
                  Intent intent_forward = new
Intent(cntx_silabasordenadas, SilabasDESordenadas.class);
      //Actividad siguiente
                  intent_forward.putExtra("letra_grupo", letra);
                  intent_forward.putExtra("color", "negro");
                  startActivity(intent_forward);
                  finish();
            }
            return true;
```

Se trata de la funcionalidad asociada al botón flecha de navegación “Siguiente actividad”. En el momento que se pulse sobre dicho botón, se ejecuta dicha parte de código. Lo que sucede es que se crea un “Intent”, llamado “intent_forward”, en dicho intent se hace referencia al contexto de trabajo actual “cntx_silabasordenadas”, y a la actividad a la que se va a saltar y que será la receptora de los datos del “Intent”, “SilabasDESordenadas.class”

```
Intent intent_forward = new Intent(cntx_silabasordenadas, SilabasDESordenadas.class);
```

después se añade a dicho “Intent” información sobre las sílabas de trabajo actuales

```
intent_forward.putExtra("letra_grupo", letra);
```

se pueden añadir más campos de información al “Intent”, en este caso se añade también información de si se va a trabaja en la actividad siguiente con silabas en color o en negro

```
intent_forward.putExtra("color", "color");
```

por último, iniciamos la nueva actividad configurada en la definición del “Intent” y finalizamos la actividad actual

```
startActivity(intent_forward);

finish();
```

En resumen, podemos decir que todas las actividades que componen la aplicación están interrelacionadas entre sí, que tenemos distintas maneras de pasar de unas actividades a otras y que se necesita realizar un intercambio de información entre actividades para mantener una coherencia durante la sesión de trabajo y que dicho intercambio de información se lleva a cabo mediante el uso de “Intents”. No importa si el salto entre actividades se realiza hacia delante o hacia atrás en la secuencia lógica de ejecución de actividades, siempre se debe producir ese flujo de información entre actividades.

6.6.1 Clase auxiliar “Metodos_aux.java”

Como ya se ha comentado, una actividad de Android está compuesta por una clase java y un “layout xml”. Sin embargo, siguiendo la filosofía de programación java puede ser interesante hacer uso de clases java auxiliares donde se pueden definir métodos (funciones de programación en java) que van a ser utilizados por las diferentes actividades. Esta clase auxiliar no lleva emparejado un “layout”, ya que no lleva asociada ninguna interfaz gráfica, es decir, una clase usada únicamente con propósitos de ayuda a otras clases de actividades.

En este caso, los métodos implementados en esta clase auxiliar serán los siguientes:

- **public static String[] crea_texto_silaba(String letra)**
  Se trata de un método público que recibe la letra o grupo de letras seleccionados en la actividad principal donde se eligen las sílabas de trabajo. Este valor es el que se pasa al método, el cual devuelve el “array de strings” con las sílabas que van a formar parte de todas las actividades hasta que se cambie de sílabas. Este “array” podría contener los valores “PA, PE, PI, PO, PU” o “GA GO GU” o “CE, CI”, por ejemplo. Todas las actividades hacen uso de éste método para obtener dichas sílabas de trabajo, por lo que

tiene más sentido implementarlo una única vez y que éste pueda ser llamado desde las actividades.

- **public static int generaAleatorio (int base)**
  Con este método se consigue generar un número aleatorio entre 0 y el número de sílabas de trabajo (4 para los casos de cinco sílabas, 2 para casos de tres sílabas y 1 para casos de dos sílabas). Todas las actividades que necesitan comenzar con una sílaba de forma aleatoria hacen una llamada a este método, veamos un ejemplo de llamada:

```
//Asignamos aleatoriamente la sílaba con la que comienza la
//actividad de las tres posibles

      vocal = Metodos_aux.generaAleatorio(total_vocales);
```

Veamos ahora la implementación del método en la clase auxiliar:

```
// Metodo que genera un numero aleatorio dentro de un rango
// dado por el valor de base

      public static int generaAleatorio(int base){

            Random r = new Random();

            int num_aleatorio = (int) r.nextInt(base);
// Generamos un numero aleatorio entre 0 y (base-1)

            return num_aleatorio;


      }
```

6.7 Base de datos SQLite

La aplicación hace uso de numerosos elementos de audio y/o imágenes en algunas de sus actividades (sin contar con los botones y otros "Views" omnipresentes durante toda la aplicación). En concreto, la cantidad de pictogramas supera los 250 en esta versión, e incluso, según se tratará en las conclusiones, en futuras versiones el volumen de pictogramas podría llegar a ser mucho mayor. Partiendo de esta premisa, parece evidente que la mejor manera de gestionar este tipo de contenidos es empleando una base de datos donde se pueda almacenar toda la información de forma centralizada y ordenada, que permita un acceso y uso sencillo de las imágenes deseadas en cada momento.

La base de datos más ampliamente usada por Android es SQLite. Sus características de ligereza y perfecta integración con Android, y por tratarse además de software libre (ver punto 4.14), la convierten en imprescindible cuando se piensa en bases de datos para Android.

La base de datos se ha creado en PC por su simplicidad en el manejo y gestión de la misma, ya que, aunque su creación y dotación de contenidos es posible directamente desde Android, la

complejidad asociada y la falta de flexibilidad para este aspecto según está diseñada la aplicación en esta versión, lo convierte en un procedimiento inviable por el momento.

Existen numerosas herramientas que permiten trabajar en PC con bases de datos SQLite, en este caso, después de probar algunas alternativas se ha optado por SQLiteBrowser (figuras 29, 30).

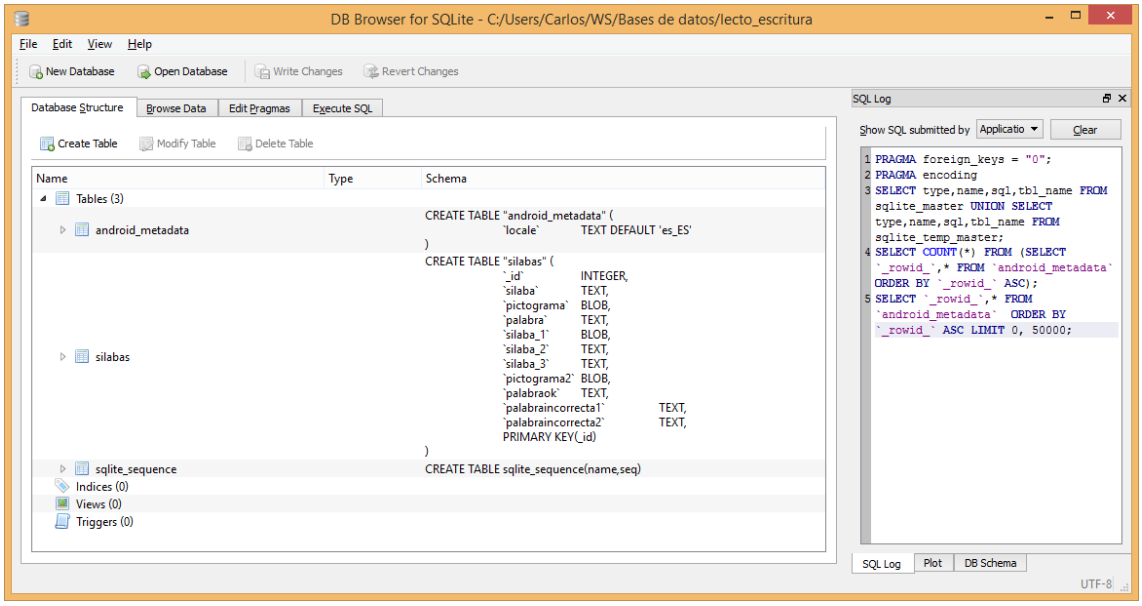


Figura 29. Interfaz de "SQLiteBrowser" mostrando la estructura de la base de datos usada en la aplicación

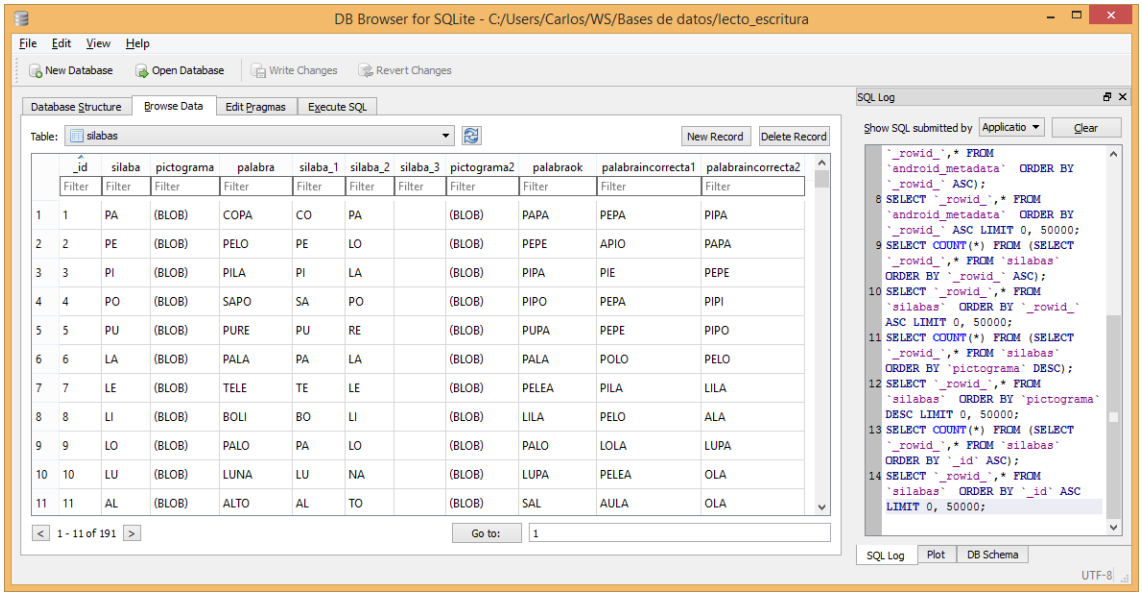


Figura 30. Interfaz de "SQLiteBrowser" mostrando contenidos de la base de datos

Una vez creada la base de datos con SQLiteBrowser, es necesario incluirla en la aplicación para que pueda ser usada por Android en tiempo de ejecución. Para ello tenemos que añadir el

fichero que contiene la base de datos (lecto_escritura) a la carpeta "assets" de la estructura de carpetas que contiene el proyecto de Eclipse (figura 31). Una vez depositada en esa carpeta, en la próxima compilación la base de datos ya formará parte de la aplicación propiamente dicha.

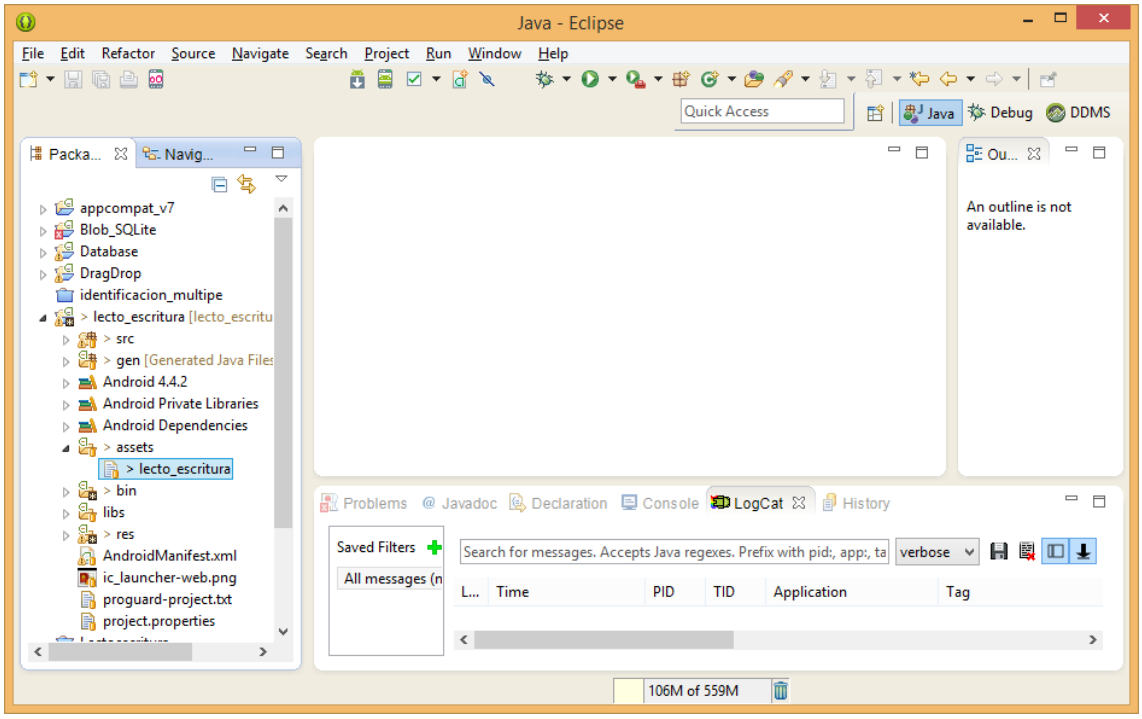


Figura 31. Base de datos "lecto_escritura" dentro de la carpeta "assets" de la aplicación

Las bases de datos en Android tienen una ubicación establecida en el directorio de ficheros que componen cada aplicación, como ya se comentó en el punto 4.14.4. En nuestro caso, la ruta absoluta donde se ubica la base de datos en la memoria del dispositivo Android es la siguiente:

```
/data/data/com.pfg_carlosgarcia.lecto_escritura/databases/nombre_basedatos
```

La gestión de la propia base de datos en la aplicación hace uso de la clase "DatabaseHelper":

```
// Creamos una instancia de la clase DBHelper para crear la base de
// datos

      DataBaseHelper myDbHelper = new DataBaseHelper(this);

// Creamos la BD si no existe, y se copia el contenido de la BD de
// la carpeta "assets" local en la carpeta de la base de datos del
// dispositivo
// /data/data/com.pfg_carlosgarcia.lecto_escritura/databases/"

            try {

                  myDbHelper.createDataBase();

            } catch (IOException e) {

                  // TODO Auto-generated catch block
                  e.printStackTrace();

                  Log.d("No existe la BD", "error");


            }
```

Una vez tenemos la instancia DBHelper, la utilizamos para recorrer la base de datos y recuperar datos mediante el uso de cursores:

```
c = myDbHelper.getRecord(silabas[vocal]);

if (c.moveToFirst()){

      silaba = c.getString(c.getColumnIndex("silaba"));

      byte[] img = c.getBlob(c.getColumnIndex("pictograma"));

      Bitmap b1=BitmapFactory.decodeByteArray(img, 0, img.length);

      String palabra = c.getString(c.getColumnIndex("palabra"));

      silaba_1 = c.getString(c.getColumnIndex("silaba_1"));

      silaba_2 = c.getString(c.getColumnIndex("silaba_2"));

      silaba_3 = c.getString(c.getColumnIndex("silaba_3"));

      //Mostrar la información en el layout
      tv_1.setText(silaba_1);

      tv_2.setText(silaba_2);

      tv_3.setText(silaba_3);

      if (silaba_2 == null || silaba_2.equals(""))

            tv_2.setVisibility(View.GONE);

      if (silaba_3 == null || silaba_3.equals(""))

            tv_3.setVisibility(View.GONE);

      //Mostramos la palabra

      tv_palabra.setText(palabra);


      //Mostramos el pictograma

      iv_picto.setImageBitmap(b1);


      // Si la silaba que hay que escribir es la primera, se ocultan
      // los otros EditText 2 y 3
      if (silaba_1.equals(silaba)){

            tv_1.setVisibility(View.GONE);

            et_2.setVisibility(View.GONE);

            et_3.setVisibility(View.GONE);


      }
```

Como vimos en la figura 29, los campos de la tabla "silabas" de la base de datos y su descripción son los siguientes:

- **_id:** clave primaria
- **Silaba:** para identificar la sílaba de trabajo
- **Pictograma:** almacena el pictograma que se corresponde con la sílaba de trabajo
- **Palabra:** palabra correspondiente al pictograma
- **Silaba1:** primera sílaba en que se descompone la palabra del campo anterior
- **Silaba2:** segunda sílaba de la palabra (si la hubiera)
- **Silaba3:** tercera sílaba de la palabra (si la hubiera)
- **Pictograma2:** almacena un segundo pictograma que se corresponde con la sílaba de trabajo
- **Palabraok:** palabra correspondiente a Pictograma2
- **Palabraincorrecta1:** palabra alternativa a la palabra correcta correspondiente a Pictograma2
- **Palabraincorrecta2:** palabra alternativa a la palabra correcta correspondiente a Pictograma2

Se ha comentado anteriormente, que esta aplicación cuenta con numerosos elementos de audio e imágenes. Las últimas ya hemos visto que se han incorporado a una base de datos para facilitar su manejo, sin embargo no así el audio. Los elementos de audio que se manejan son las locuciones de las sílabas, instrucciones de las actividades y algunos refuerzos sonoros para los aciertos o errores cometidos.

Al contrario que con las imágenes, donde es previsible un incremento considerable en el número de ellas (pictogramas) en futuras versiones, los elementos de audio no parece que vayan a contar con esa tendencia. Por ello se ha decidido excluirlos de la base de datos, pero además, el manejo de locuciones es más ágil con el acceso directamente a ficheros que a través de una base de datos, que siempre añade pasos adicionales para la recuperación de la información.

Existe un fichero por cada una de las sílabas de la aplicación. Estos ficheros de audio han sido nombrados con la transcripción de su sílaba, y con la extensión ".mp3", salvo algunas excepciones que se comentarán en el siguiente apartado. La ruta donde se almacenan los ficheros de audio se puede ver en la figura 32, así como algunos de los ficheros.

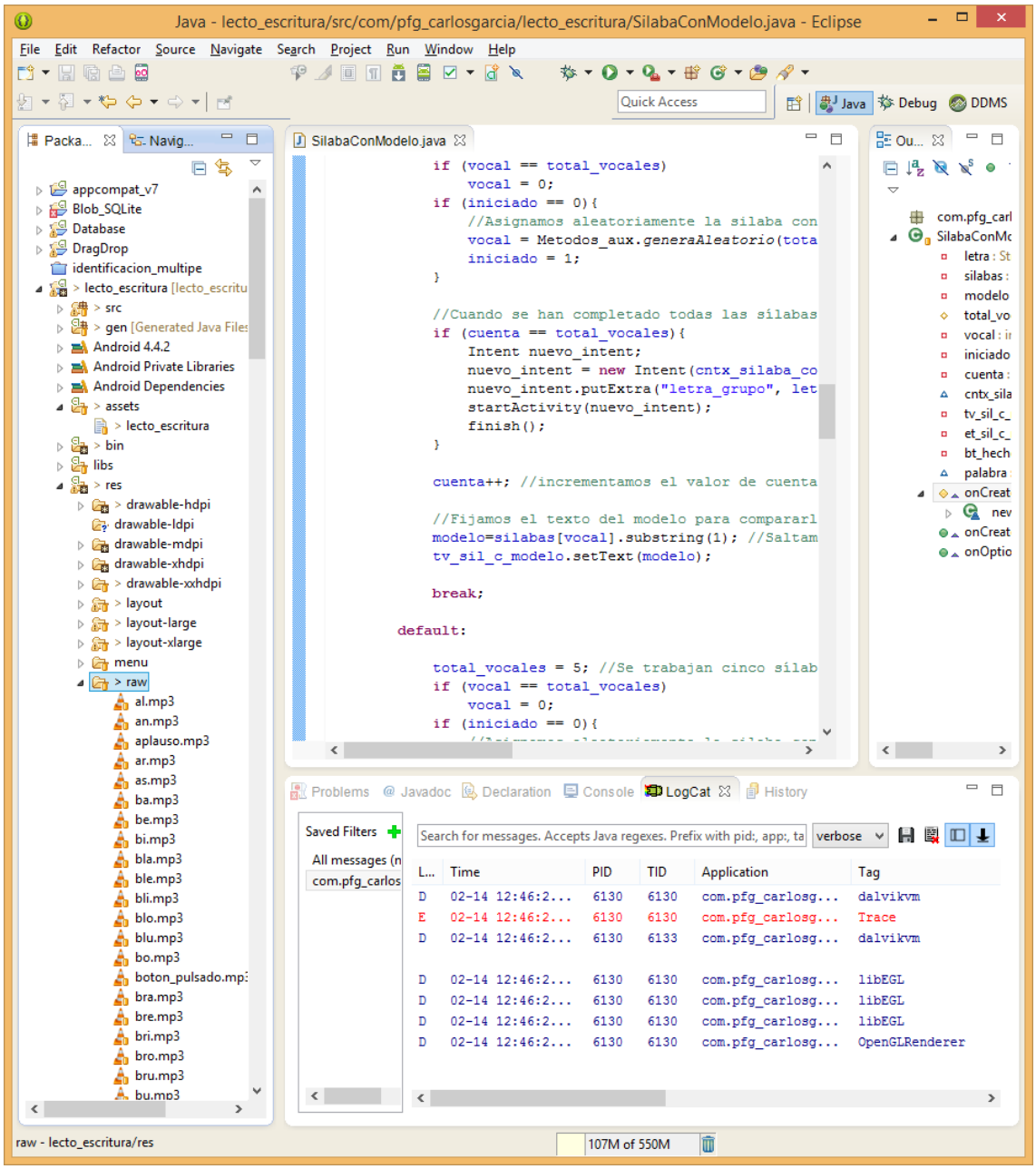


Figura 32. Ficheros de locuciones

Las locuciones han sido grabadas y procesadas por el autor de la aplicación, a partir de voces de dos niños que han prestado su colaboración para esta labor. En un principio se pensó usar un sintetizador de voz por simplicidad y comodidad, pero a petición de los logopedas del colegio colaborador se optó por voces reales de niños para hacer más atractiva la aplicación al público objetivo de la misma.

6.8 Situaciones singulares de sílabas

Se ha podido apercibir en apartados anteriores, que no existe una regularidad en la cantidad de sílabas de trabajo, ni en el número de letras que componen cada sílaba. Podemos decir, en

general que todas las sílabas cuenta con dos o tres letras por sílaba. Tras esa primera clasificación podemos hacer otra clasificación más profunda por el número de sílabas de trabajo en total:

- **Dos sílabas:** se trata de las situaciones que se presentan cuando se va a trabajar con los siguientes grupos de sílabas: **"QUE, QUI"**, **"GUE, GUI"**, **"GE, GI"**, **"CE, CI"**
- **Tres sílabas:** se trata de las situaciones que se presentan cuando se va a trabajar con los siguientes grupos de sílabas: **"CA, CO, CU"**, **"GA, GO, GU"**, **"ZA, ZO, ZU"**
- **Cinco sílabas:** el resto de casos.

Dentro del grupo de cinco sílabas volvemos a encontrar situaciones excepcionales que requieren un tratamiento específico para cada una de ellas:

- **"R" suave:** dado que su representación gráfica coincide exactamente con la de la "R" fuerte, pero su sonido es diferente ("ca**RO**", frente a "**RO**to", por ejemplo), se ha establecido la convención de anteponer un guion a la sílaba suave ("-RO"). Así pues, en la actividad de presentación podremos encontrar las diferencias según vemos en las figuras 33 y 34. Otra peculiaridad es que, como ya hemos dicho, los nombres de ficheros de audio son la transcripción de la sílaba correspondiente. En este caso tendríamos dos ficheros con diferente audio y con el mismo nombre. Para solventar esta situación, se ha optado por mantener el criterio inicial para la "R" fuerte (ra.mp3), y añadir el sufijo "_s" para la "R" suave (ra_s.mp3). Estas situaciones respecto al nombre de los ficheros de audio, así como el uso del prefijo guion deben ser tenidas en cuenta en las actividades, tanto si hacen uso del fichero de audio, o si solo gestionan la representación visual de la sílaba, o incluso ambos casos al tiempo.

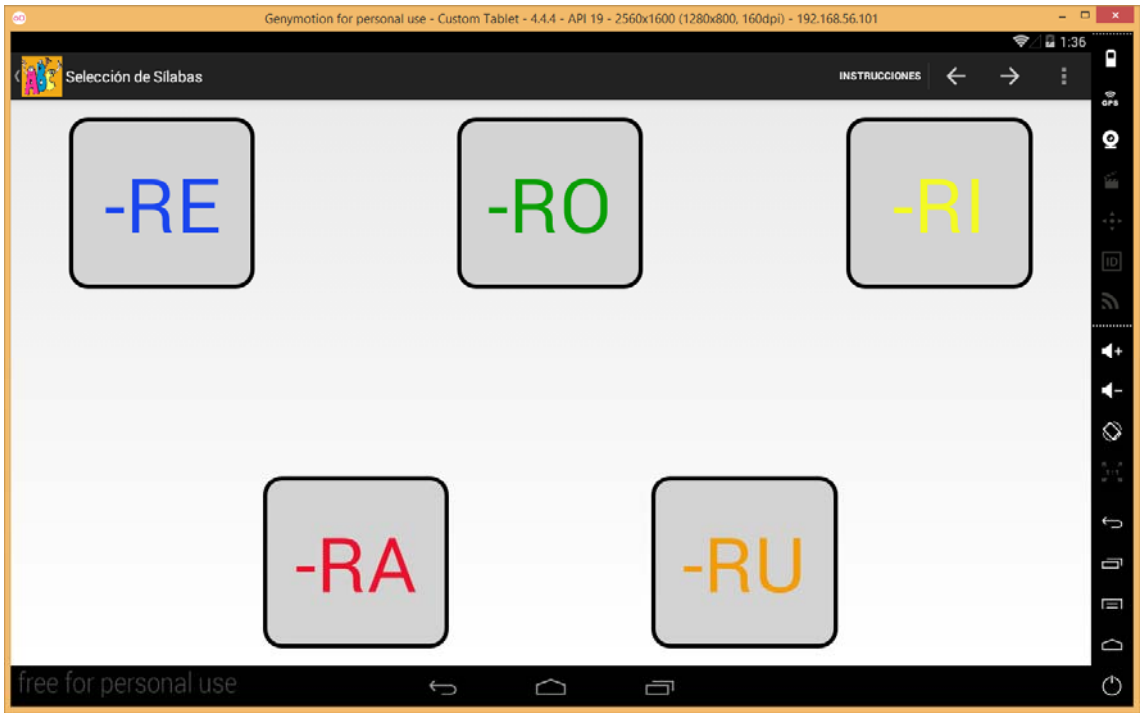
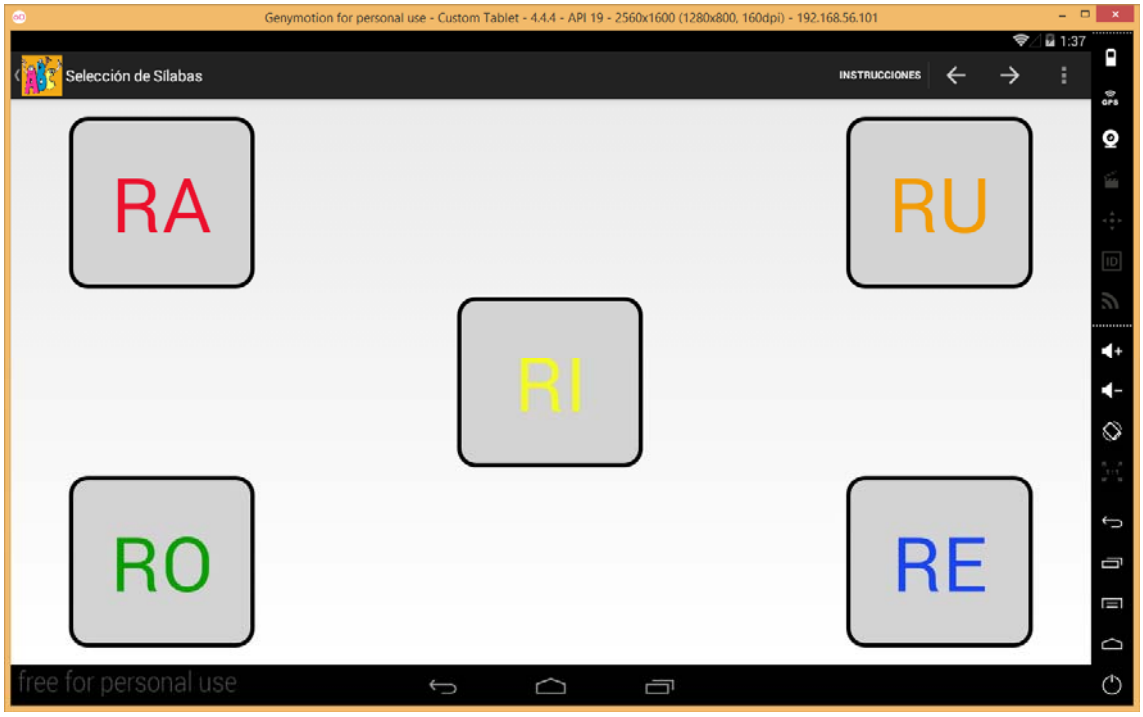


Figura 33. Representación gráfica de las sílabas "R" suave



34. Representación gráfica de las sílabas "R" fuerte

- **"R" fuerte doble (RR):** se trata de sílabas regulares, cuya única particularidad es que su sonido es el mismo que el de la "R" fuerte sencilla. Por ello, en las actividades en que se reproduce su audio, se debe controlar dicha situación de forma excepcional, y utilizar el sonido de su sílaba equivalente con "R" fuerte sencilla. En la figura 35 se plasma la representación gráfica para tener una visión completa de las posibilidades de formación de sílabas con la letra "R".

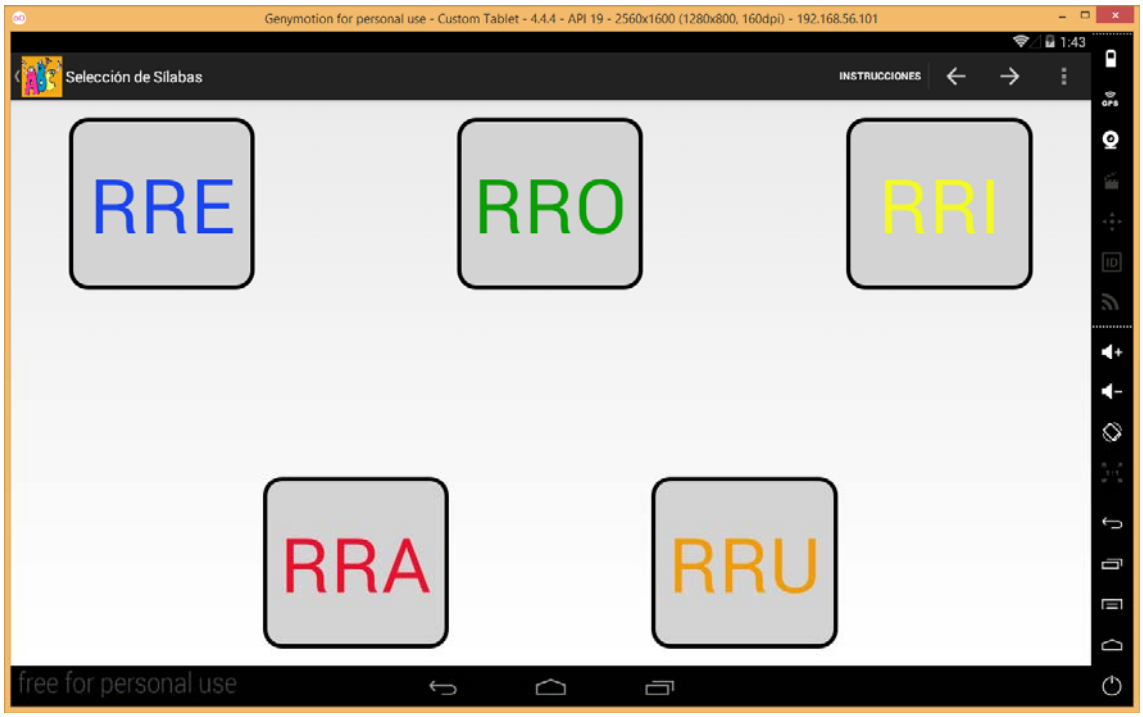


Figura 35. Representación gráfica de las sílabas "R" fuerte doble

- **Ñ:** Se trata de un grupo de sílabas regulares, el único inconveniente que presenta es que, como ocurre a menudo en tecnologías de la información, los nombres de fichero que contienen la letra "Ñ" pueden ocasionar problemas en sistemas Android, por lo que se ha decidido utilizar una nomenclatura específica sustituyendo la letra "ñ" por las letras "gn" (gna.mp3 en lugar de ña.mp3). Figura 36.

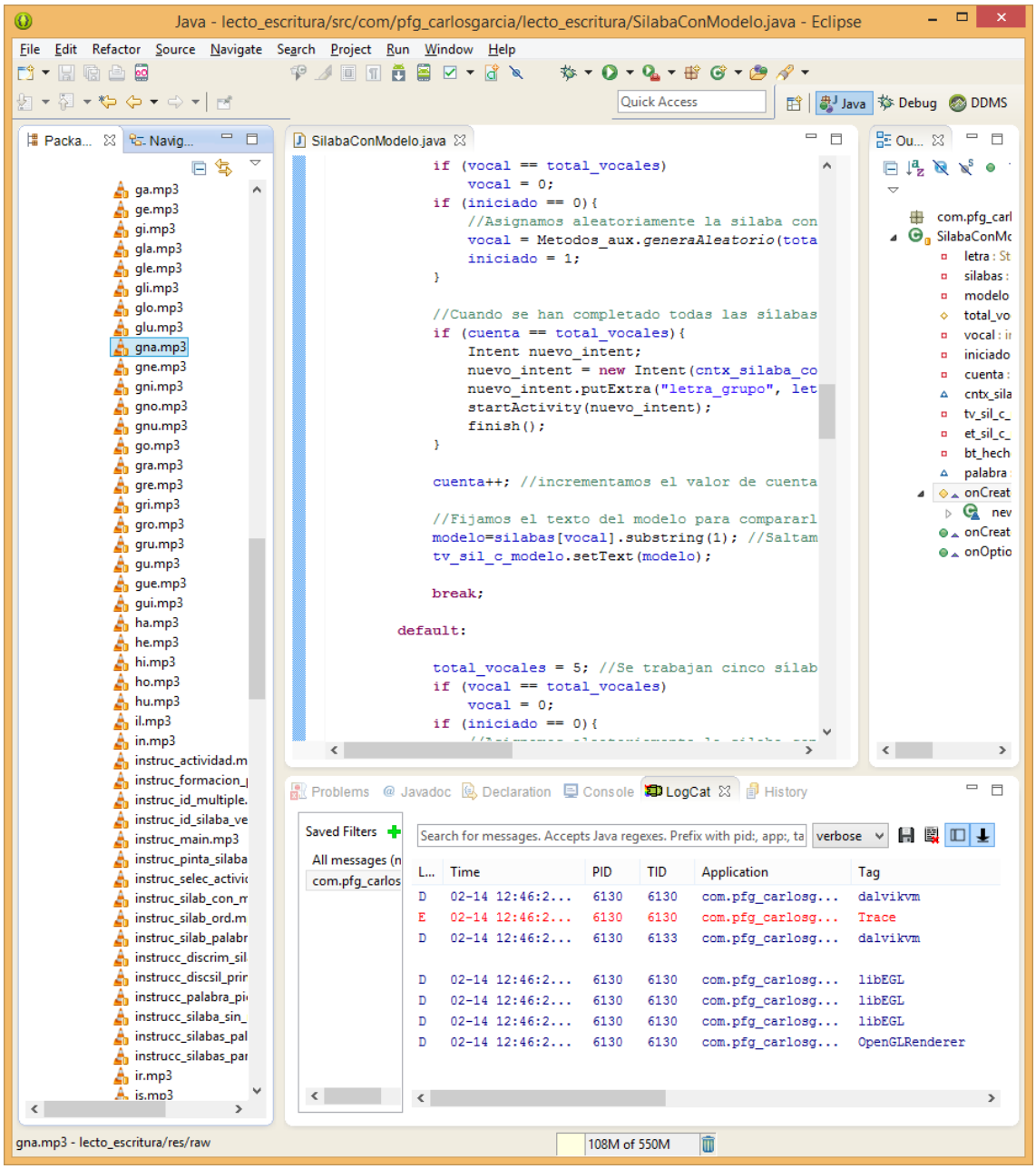


Figura 36. Nombres de fichero para locuciones de silabas con "Ñ"

- **FL:** se trata de un grupo de sílabas completamente regular de cinco sílabas tanto en su representación gráfico como en audio. Su única singularidad reside en que no se ha podido encontrar una palabra que contenga la sílaba **"FLI"** cuyo pictograma pueda ser comprendido fácilmente por los niños. La singularidad solo es perceptible en actividades que visualicen pictogramas aunque no es algo demasiado notorio. Es la única excepción de este tipo que se ha producido en el desarrollo de la aplicación.

Estas irregularidades son propias del lenguaje castellano, y para manejarlas correctamente es necesario crear estructuras de control de programación que gestionen adecuadamente estas situaciones irregulares. Veamos un ejemplo de cómo se manejan cada una de las situaciones

anteriormente detalladas haciendo uso de las sentencias “switch-case” del lenguaje de programación java.

Para los conjuntos de tres silabas tendríamos algo así:

```
switch(letra){

case "CA CO CU":
case "GA GO GU":
case "ZA ZO ZU":

      total_vocales = 3; //Se trabajan tres sílabas


      // Si se ha alcanzado la última vocal, pero no se han
      // completado todas las sílabas, pasamos a la vocal 0

      if (vocal == total_vocales)


                        vocal = 0;

-

-

-
```

Para los grupos de dos sílabas:

```
case "QUE QUI":
case "GUE GUI":
case "GE GI":
case "CE CI":

      total_vocales = 2; //Se trabajan dos sílabas


      // Si se ha alcanzado la última vocal, pero no se han
      // completado todas las sílabas, pasamos a la vocal 0


      if (vocal == total_vocales)

            vocal = 0;

-

-

-

-
```

De manera similar procederíamos con el resto de situaciones excepcionales, y concluiríamos las sentencias de control “switch-case” con las acciones para los grupos de sílabas completamente regulares, haciendo uso de la estructura de control “default”, que es el bloque de código que se ejecuta cuando no se ha cumplido ninguno de los case anteriores:

```
default:

      total_vocales = 5; //Se trabajan cinco sílabas
      // Si se ha alcanzado la última vocal, pero no se han
      // completado todas las sílabas, pasamos a la vocal 0

      if (vocal == total_vocales)

            vocal = 0;

-

-

-
```

7. Conclusiones

Podemos resumir que se ha desarrollado una aplicación Android completamente funcional cuyo objetivo principal ha sido cubrir un vacío en el mundo de las aplicaciones para dispositivos móviles orientadas a la enseñanza de la lectoescritura para niños con discapacidad. Se ha partido de unos requisitos que debería cumplir esta aplicación a partir de los cuales se han ido desarrollando las diferentes actividades que la componen. Se han utilizado técnicas de programación para sistemas Android, con lenguaje de programación Java, así como emuladores para prueba y depuración junto con dispositivos físicos reales para mayor fiabilidad.

El desarrollo en conjunto del proyecto ha sido muy laborioso en las tareas de aprendizaje e investigación, pero sobre todo, debido a la gran cantidad de actividades implementadas (veinticuatro en total), que aunque si bien en muchos casos ha permitido reutilizar el código desarrollado en actividades anteriores, también es cierto que ha sido necesario hacer adaptaciones continuas debido a las muchas singularidades y falta de regularidad en el uso de sílabas y pictogramas. Hay que añadir que la mayor parte de los contenidos de audio han sido creados por el autor de la aplicación con sus propios medios. También ha sido necesario crear y dotar de contenidos una base de datos para el correcto desempeño de las actividades con una gran cantidad de pictogramas y palabras asociadas.

Por el camino han aparecido dificultades muy específicas, sobre todo en forma del desarrollo de determinadas funcionalidades necesarias, difícilmente localizables en bibliografía general, y que han sido resueltas a menudo en foros de sitios web de desarrollo, gracias a la actividad de otros programadores que han pasado por situaciones similares y han plasmado su conocimiento de forma pública y accesible.

El resultado final parece ser satisfactorio ya que se ha probado en un colegio con niños con necesidades educativas especiales resultando en una estupenda acogida. Los niños han mostrado gran interés y entusiasmo y los profesionales de dicho colegio han dado su aprobación a la aplicación. Previsiblemente el uso de esta herramienta, será introducida paulatinamente en sus rutinas de trabajo habituales.

Como se ha comentado, se ha utilizado una base de datos para dotar de una capacidad extra de flexibilidad. Sin embargo en el estado actual de la aplicación, esa flexibilidad solo ha sido palpable de cara a la obtención de recursos directamente de la base de datos, en lugar de estar manejando ficheros que podría llegar a ser una actividad más engorrosa cuando el número de éstos es de cierta envergadura. Es decir, la base de datos se ha creado en un ordenador con todos sus contenidos y después se ha incorporado a la aplicación para que pueda ser gestionada por ésta. Sin embargo no se ha dotado a la aplicación de la posibilidad de ampliar dicha base de datos.

La ampliación de la base de datos desde la propia aplicación no es una tarea sencilla, y podría ser una buena oportunidad para futuros PFGs, la posibilidad de continuar la labor que ahora se termina, con nuevas versiones de esta herramienta que pongan el foco en esta funcionalidad. Sería también interesante, la posibilidad de utilizar bases de datos externas, gestionadas de forma personal desde los propios PCs. Otra posibilidad más compleja, pero sin duda más interesante, sería disponer de repositorios públicos de bases de datos (estableciendo posibles vínculos con organizaciones relacionados con los SAAC, por ejemplo). De esta manera, se

podrían descargar bases de datos directamente desde la aplicación, a través de interfaces web, lo que proporcionaría un potencial mucho mayor para los fines perseguidos.

Otra posibilidad para continuar con labor actual, sería la opción de trabajo en forma “Terapeuta-Alumno”. El terapeuta (o cualquier profesional que guíe al niño en el uso de la herramienta), programaría una serie de actividades que el niño tendría que realizar. El resultado del trabajo del niño sería registrado en una base de datos que podría ser local o remota a través de Internet. Los registros con los resultados de las distintas sesiones de trabajo de los niños podrían ser posteriormente analizados y comparados en el tiempo para comprobar los progresos obtenidos en un determinado periodo de tiempo. Se podrían realizar también estadísticas y otros estudios con los datos registrados.

8. Bibliografía

http://developer.android.com/index.html

http://stackoverflow.com/

http://es.wikipedia.org

http://commons.wikimedia.org/

http://www.androidcurso.com/

http://www.google.es

https://eclipse.org/

https://www.genymotion.com

https://www.sqlite.org/

http://sqlitebrowser.org/

http://developer.android.com/tools/studio/index.html

http://www.who.int/es/

http://www.catedu.es/arasaac/index.php

9. Referencias

[1] Handset Alliance [disponible on-line: http://www.openhandsetalliance.com], consultado en enero de 2015

[2] IDC [disponible on-line: http://www.idc.com/prodserv/smartphone-os-market-share.jsp], consultado en enero de 2015

[3] Plataforma informática [disponible on-line: http://es.wikipedia.org], consultado en enero de 2015

[4] OMS, Discapacidades [disponible on-line: http://www.who.int/topics/disabilities/es/], consultado en febrero de 2015

[5] Ley Orgánica 2/2006, de 3 de mayo, de Educación. Texto consolidado [disponible on-line: http://www.boe.es/buscar/pdf/2006/BOE-A-2006-7899-consolidado.pdf], consultado en febrero de 2015

[6] Gartner Says Worldwide Tablet Sales Grew 68 Percent in 2013, With Android Capturing 62 Percent of the Market [disponible on-line: http://www.gartner.com/newsroom/id/2674215], consultado en febrero de 2015

[7] Android Developers Dashboards [disponible on-line: https://developer.android.com/about/dashboards/index.html], consultado en febrero de 2015

[8] Diccionario de la lengua española – RAE [disponible on-line: http://lema.rae.es/drae/?val=lectoescritura], consultado en febrero de 2015

Anexo I. Sílabas de trabajo

Se pueden clasificar a sílabas en castellano de la siguiente manera:

- **Directas**: son la combinación Consonante + Vocal (CV), PA, ME, LO, SU…
- **Inversas:** son la combinación Vocal + Consonante (VC), AL, EN, IR, OS…
- **Trabadas:** combinación Consonante + Consonante + Vocal (CCV), BLA, PLE, BRO, TRU…
- **Mixtas:** combinación CVC, como por ejemplo "CIN" de "CINtura" o "LEN" de "LENtejas" o "PAN" de "PANtano"... Como las combinaciones son muchísimas, sería imposible reflejarlas todas.

De cualquier modo, si un niño sabe leer "CI" y también sabe leer "IN" va a saber cómo leer CIN, por eso no es necesario enseñar estas sílabas de forma sistemática. Resumiendo, al trabajar las sílabas directas y las inversas no es necesario enseñar una a una las mixtas porque son combinación de las anteriores.

Por otro lado, se ha obviado el uso de acentos ortográficos en las sílabas, ya que pertenecen a un nivel educativo superior al del ámbito de esta aplicación.

Se listan a continuación todas las sílabas (Directas, Inversas y Trabadas) que se trabajan en la aplicación, por orden de aparición en la pantalla principal (cuadrícula de 7 filas por 6 columnas).

- **FILA 1**

PA	PE	PI	PO	PU
LA	LE	LI	LO	LU
AL	EL	IL	OL	UL
SA	SE	SI	SO	SU
AS	ES	IS	OS	US
TA	TE	TI	TO	TU

- **FILA 2**

NA	NE	NI	NO	NU
AN	EN	IN	ON	UN
MA	ME	MI	MO	MU
DA	DE	DI	DO	DU
BA	BE	BI	BO	BU
JA	JE	JI	JO	JU

- **FILA 3**

ÑA ÑE ÑI ÑO ÑU

FA FE FI FO FU

CA CO CU

QUE QUI

LLA LLE LLI LLO LLU

RA RE RI RO RU

- **FILA 4**

RRA RRE RRI RRO RRU

AR ER IR OR UR

GA GO GU

GUE GUI

GE GI

HA HE HI HO HU

- **FILA 5**

ZA ZO ZU

CE CI

RA RE RI RO RU **(suave)**

CHA CHE CHI CHO CHU

VA VE VI VO VU

YA YE YI YO YU

➢ **FILA 6**

BLA	BLE	BLI	BLO	BLU
BRA	BRE	BRI	BRO	BRU
PLA	PLE	PLI	PLO	PLU
PRA	PRE	PRI	PRO	PRU
CLA	CLE	CLI	CLO	CLU
CRA	CRE	CRI	CRO	CRU

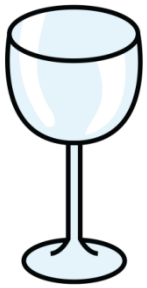
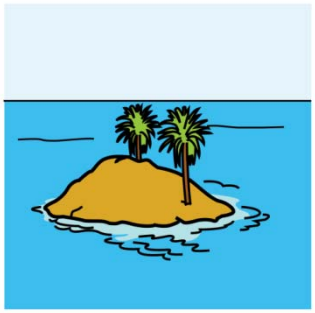
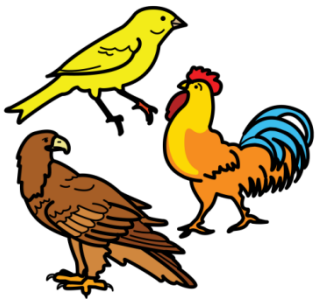
➢ **FILA 7**

FLA	FLE	FLI	FLO	FLU
FRA	FRE	FRI	FRO	FRU
GLA	GLE	GLI	GLO	GLU
GRA	GRE	GRI	GRO	GRU
TRA	TRE	TRI	TRO	TRU
DRA	DRE	DRI	DRO	DRU

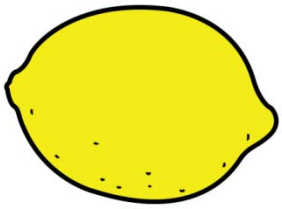
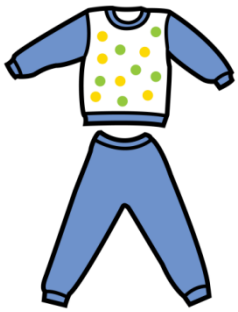
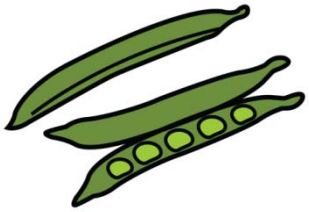
Anexo II. Pictogramas. Condiciones de uso.

Se muestran a continuación algunos ejemplos de los pictogramas usados en la aplicación. Los símbolos pictográficos utilizados de ARASAAC (http://catedu.es/arasaac/)[1] son parte de una obra colectiva propiedad de la Diputación General de Aragón y han sido creados bajo licencia Creative Commons (BY-NC-SA)[2] por Sergio Palao.

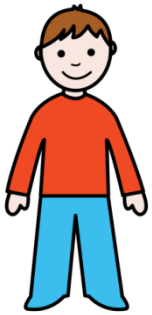
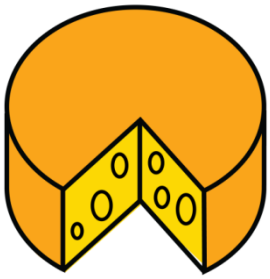
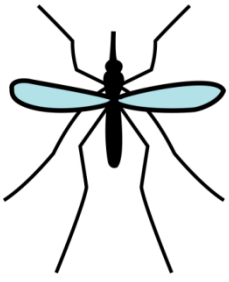
FILA 1

 PELO	 COPA
 ISLA	 PAJAROS

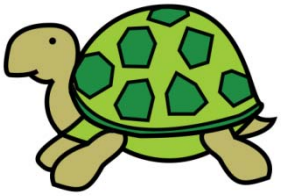
FILA 2

 MIEL	 LIMON
 PIJAMA	 JUDIAS

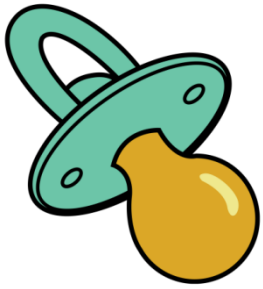
FILA 3

 ALBAÑIL	 NIÑO
 QUESO	 MOSQUITO

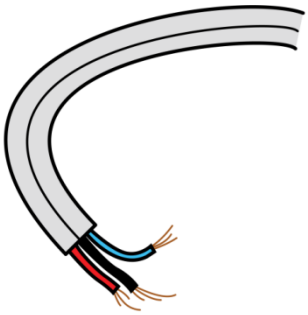
FILA 4

 ARRUGAR	 CARRITO
 TORTUGA	 GORRO

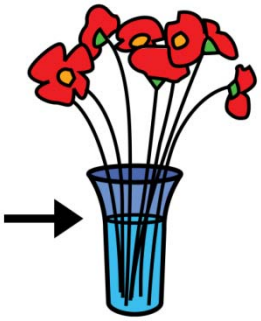
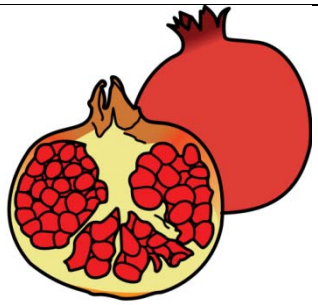
FILA 5

 BOMBERO	 SIRENA
 CHUPETE	 LECHE

FILA 6

 BLOQUES	 CABLE
 CROMOS	 CREMA

FILA 7

 FLORERO	 FLAN
 GRANADA	 GRUPO

9.1 Referencias Anexo II

[1] Portal Aragonés de la Comunicación Aumentativa y Alternativa [disponible on-line: http://catedu.es/arasaac/], consultado en septiembre de 2014

[2] Creative Commons [disponible on-line: http://creativecommons.org/licenses/by-nc-sa/3.0/es/], consultado en febrero de 2015

Anexo III. Genymotion

Genymotion es un emulador Android sencillo rápido y potente. Es una herramienta de valor inestimable para probar y depurar aplicaciones desarrollada en entorno Android. Presenta una interfaz de usuario amigable y sencilla, que la hace muy atractiva para el trabajo diario en el mundo de la programación Android

Una de sus principales bazas es que se integra perfectamente en IDEs como Eclipse o Android Studio, por medio de sus correspondientes "plugins". Además presenta unas características avanzadas como la emulación de sensores (GPS, micrófono, cámara, batería, multi táctil, y acelerómetro), posibilidad de arrastrar y soltar ficheros desde el PC hasta el dispositivo emulado (por ejemplo, para la instalación de aplicaciones con fichero ".apk", basta con arrastrar dicho fichero a la pantalla emulada y el proceso de instalación comenzará automáticamente). Incluso puede hacer uso del portapapeles del ordenador para pegar información en el dispositivo Android.

Otras opciones disponibles son la posibilidad de redimensionar las pantallas, posibilidad de capturas de pantalla, aceleración gráfica, virtualización de "CPUs" e incluso una herramienta de línea de comandos, aunque esta opción no se ha usado para el desarrollo de este proyecto.

Ofrece por defecto veinte dispositivos pre configurados, más la posibilidad de crear dispositivos propios a media.

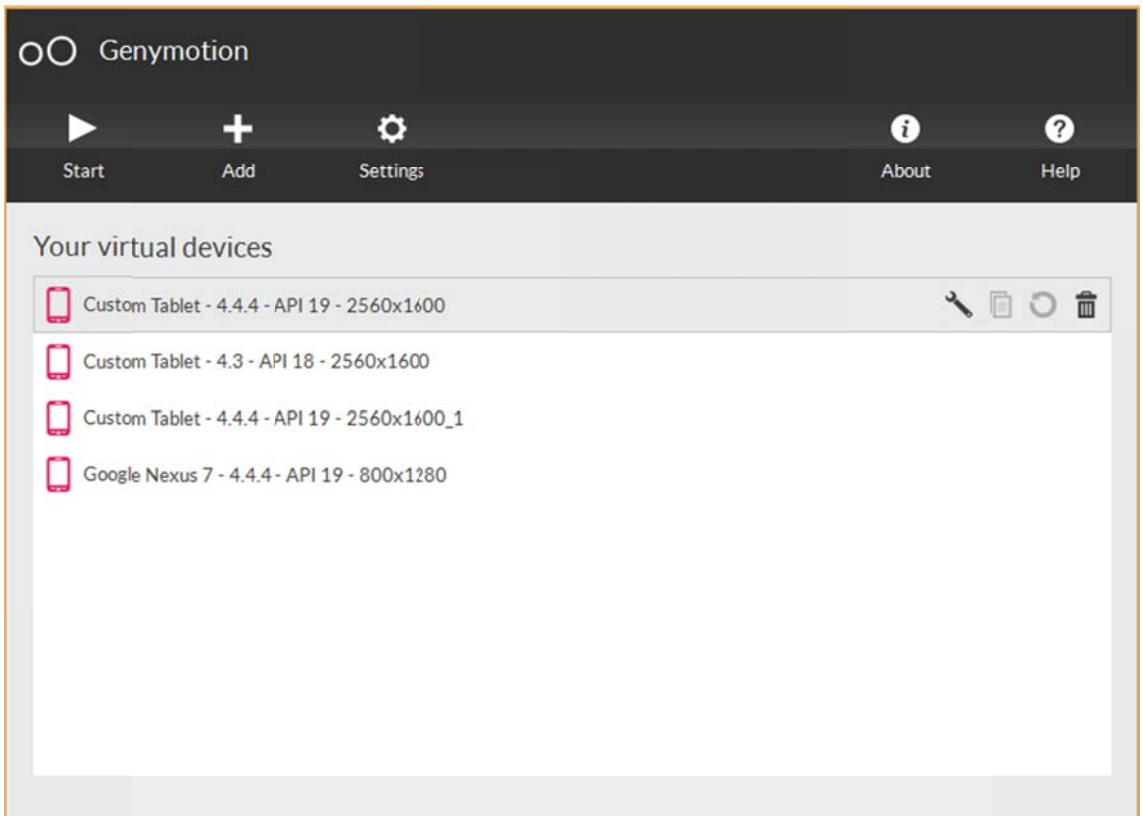


Figura 37. Dispositivos disponibles para el desarrollo de este proyecto

Figura 38. Aspecto de uno de los dispositivos en ejecución

Anexo IV. Manual de instrucciones

Con esta aplicación se pretende facilitar la enseñanza de la lectoescritura (enseñanza y aprendizaje de la lectura simultáneamente con la escritura) orientada a niños con necesidades educativas especiales.

La aplicación está divida en grupos de actividades, que a su vez se componen de diferentes actividades cada grupo.

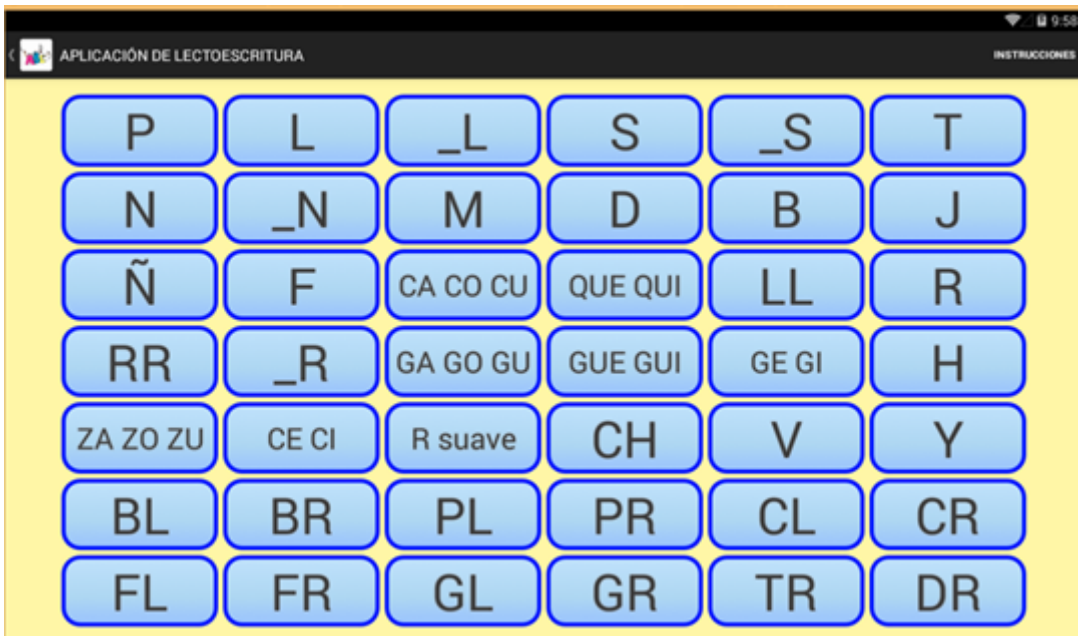
Requisitos de sistema

La aplicación está especialmente orientada a dispositivos Android con pantallas de siete pulgadas o mayores. Dadas las características visuales de la misma, no se recomienda su uso en dispositivos con pantallas menores, si bien su funcionamiento es posible, y dependiendo de las características del mismo el resultado final puede ser más o menos satisfactorio.

La versión mínima de software será Android 3.0 (nivel de API 11). Este requisito es mandatorio, si no se cumple, la aplicación no se ejecutará.

Pantalla principal

En la primera pantalla que aparece al arrancar la aplicación, encontraremos una serie de botones que nos permitirán elegir la letra o letras que nos llevarán a las sílabas de trabajo deseadas (figura 39).



39. Pantalla principal

El uso lógico por motivos pedagógicos de la aplicación sería en primer lugar elegir las sílabas de trabajo, comenzando por el primer botón (de arriba hacia abajo y de izquierda a derecha, es

decir comenzar por las sílabas “PA, PE, PI, PO, PU”), y una vez realizadas todas las actividades de todos los grupos referidas a dichas sílabas, continuar con el siguiente botón (“LA, LE, LI, LO, LU”) y con sus actividades correspondientes. El orden de trabajo de grupos de actividades se debe realizar siguiendo el orden en que está estructurado el menú de la figura 18, desde arriba hacia abajo y de izquierda a derecha, criterio que se usará de igual modo para las actividades dentro de cada grupo.

En resumen, habría que seguir el esquema de actividades citado un poco más arriba en orden descendente, trabajando las sílabas en el orden también citado en el párrafo anterior.

Grupos de actividades

Tras seleccionar las sílabas que se van a trabajar, se nos presenta un menú con cinco posibilidades (cinco grupos de actividades):

- Presentación
- Discriminación visual
- Discriminación auditiva
- Lectura
- Escritura

Dentro de cada grupo se presentarán distintas actividades que permitirán el trabajo objetivo del grupo desde diferentes puntos de vista, o simplemente alternando el uso de colores en las sílabas o el uso solo del color negro. Respecto al color de las sílabas se ha seguido un código de colores fijo para todas las actividades. Las sílabas que se forman con la vocal “A” serán de color rojo, el azul será para la “E”, amarillo para la “I”, verde para la “O” y naranja para la “U”.

Se detallan a continuación los grupos y sus actividades:

- **PRESENTACIÓN**
  - Sílabas ordenadas color
  - Sílabas desordenadas color
  - Sílabas ordenadas negro
  - Sílabas desordenadas negro

- **DISCRIMINACIÓN VISUAL**
  - Identificación sílabas color
  - Identificación sílabas negro
  - Identificación múltiple color
  - Identificación múltiple negro
  - Identificación parejas color
  - Identificación parejas negro
  - Identificación en palabra color
  - Identificación en palabra negro

- **DISCRIMINACIÓN AUDITIVA**
  - Discriminación sílabas color
  - Discriminación sílabas negro

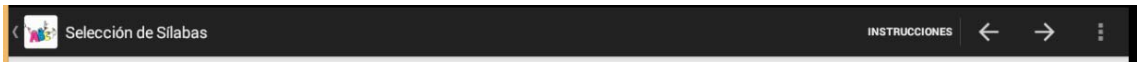
- Discriminación sílabas color en palabra
- Discriminación sílabas negro en palabra

➢ **LECTURA**
- Asociación palabra pictograma

➢ **ESCRITURA**
- Pinta la sílaba
- Escribe la sílaba con modelo
- Escribe la sílaba sin modelo
- Escribe la sílaba en palabra con modelo
- Escribe la sílaba en palabra sin modelo
- Formación de palabras

Navegación por la aplicación. Instrucciones

Para movernos por la aplicación, entre actividades existen dos opciones. La primera es una vez seleccionadas las sílabas de trabajo, ir accediendo a los distintos menús presentes en la aplicación. La otra es comenzar con una actividad y una vez finalizada pasar a la siguiente o a la anterior haciendo uso de las flechas de navegación adelante y atrás de la barra de acciones, como vemos en la figura 40.



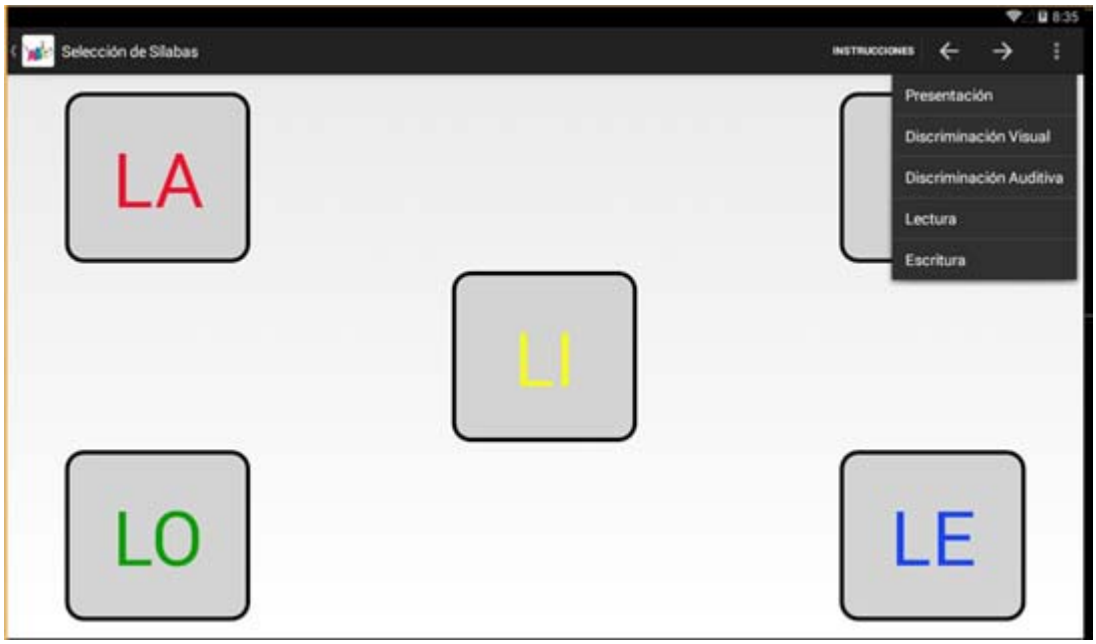
40. Barra de acciones

Con estas flechas pasamos de actividad sin tener en cuenta si se cambia el grupo de actividades o no. Por ejemplo, cuando estamos en la última actividad del grupo “Presentación”, “Sílabas desordenadas negro”, y pulsamos sobre la flecha hacia la derecha (siguiente actividad), pasaremos directamente a la primera actividad de “Discriminación visual”, es decir, a la actividad “Identificación sílabas color”. En todo caso se van arrastrando las sílabas de trabajo actuales entre actividades. Casos particulares de navegación son la primera actividad “Sílabas ordenadas color”, que solo presenta la flecha de navegación de actividad siguiente, y la última actividad, “Formación de palabras”, que solo presenta la flecha de navegación de actividad anterior. De este modo se impide un comportamiento circular de navegación, especificando con claridad cuál es el comienzo y cuál es el fin de las actividades de unas sílabas de trabajo.

Para facilitar la compresión de los objetivos de las diferentes partes de la aplicación, se ha incluido en la barra de acciones, un botón “INSTRUCCIONES” (ver figura 39), que una vez pulsado presentará un mensaje en pantalla con una breve explicación de la actividad y simultáneamente se reproducirá una locución con el mismo mensaje.

Dentro de la barra de acciones encontramos también el icono de la aplicación a la izquierda de la misma (figura 21). En cualquier momento y en cualquier actividad si pulsamos sobre el icono, nos moveremos a la pantalla principal y podremos cambiar las sílabas de trabajo.

Por último, en el extremo derecho de la barra (figura 40), podemos ver tres puntos alineados en vertical. Pulsando aquí tendremos acceso a un menú donde podremos elegir el grupo de actividades de los cinco posibles y que ya se han mencionado anteriormente (figura 41).



41. Menú de grupos en barra de acciones

Orden de trabajo de las sílabas

El orden de trabajo de las sílabas dentro de una actividad se ha llevado a cabo de forma aleatoria, es decir, que si las sílabas de trabajo son “PA, PE, PI, PO, PU”, no sea siempre la sílaba “PA” la que comience una actividad, sino cualquiera de las cinco. De este modo se evita que los niños aprendan que siempre la primera sílaba de trabajo será la formada con la vocal “A”, evitando que se guíen por una sucesión lógica. Esta forma de presentar las sílabas no afecta a algunas actividades en las que carece de sentido, como por ejemplo cuando se presenta todas las sílabas (Actividades de “Presentación”).